N° d'enregistrement

au C.N.R.S.

THESE DE DOCTORAT D'ETAT ES SCIENCES : Mathématiques

présentée à

L'UNIVERSITE PARIS VI

par

M. Alain CHECROUN

pour obtenir

le grade de Docteur Es-Sciences

Sujet de la Thèse : Traitement et Reconnaissance des Formes par un Ordinateur. Application à la Cartographie Electromagnétique.

Soutenue le 4 Mars 1972 devant la Commission d'Examen

MM.	J. L. LIONS	Président
	P. AIGRAIN	Examinateurs
	J. C. SIMON	
MM.	BARDOS	Invités
	BERTHET	

N° d'enregistrement

au C.N.R.S.

THESE DE DOCTORAT D'ETAT ES SCIENCES : Mathématiques

présentée à

L'UNIVERSITE PARIS VI

par

M. Alain CHECROUN

pour obtenir

le grade de Docteur Es-Sciences

Sujet de la Thèse : Traitement et Reconnaissance des Formes par un Ordinateur. Application à la Cartographie Electromagnétique.

Soutenue le 4 Mars 1972 devant la Commission d'Examen

MM.	J. L. LIONS	Président
	P. AIGRAIN	Examinateurs
	J. C. SIMON	
MM.	BARDOS	Invités
	BERTHET	

UNIV.
LEIDEN
BIBL.

Je remercie vivement Monsieur le Professeur LIONS de m'avoir fait l'honneur de présider le jury de cette thèse ainsi que Messieurs les Professeurs AIGRAIN, BARDOS et BERTHET d'avoir bien voulu constituer le jury.

Que le Professeur J. C. SIMON trouve ici l'expression de ma gratitude pour les conseils et encouragements qui ont permis de faire aboutir ce travail.

Madame RIVIERE a eu la lourde tache de la dactylographie et Madame HANNEQUIN et son équipe ont réalisé le tirage et la reliure. Je les remercie de leur gentillesse et de l'excellent travail qu'elles ont fait.

Digitized by the Internet Archive
in 2026

https://archive.org/details/IA41549225_0003

TRAITEMENT ET RECONNAISSANCE DES FORMES PAR UN ORDINATEUR

Application à la Cartographie Electromagnétique (RADAR)

C H A P I T R E I

Formalisation et Langage adaptés aux processus de traitement sur Ordinateur des signaux et images.

§1. LE PROBLEME DE LA RECONNAISSANCE DES FORMES PAR UN ORDINATEUR

1.1. Définition.

1.2. Représentation des formes dans un ordinateur.

1.3. Les différentes étapes de la reconnaissance d'une forme.

§2. PRISE ET CONSERVATION DE L'INFORMATION - ECHANTILLONNAGES

2.1. L'information d'une image.

2.2. Utilisation d'un support borné.

2.3. Utilisation d'échantillons discrets.

2.4. Exemple - Application à la représentation des lignes.

C H A P I T R E II

Traitement des images

§1. TRANSFORMATIONS SUR UNE REPRESENTATION

1.1. Transformations ponctuelles.

1.2. Transformations linéaires.

1.3. Filtres digitaux.

§2. LE PROBLEME DU RECALAGE D'UN AVION A PARTIR D'INFORMATIONS RADAR

2.1. Les données actuelles.

2.2. Application du filtrage digital pour réduire le volume de ces données.

§3. EXTRACTION DES PRIMITIVES DANS LE CAS D'IMAGES FORMEES DE LIGNES

3.1. Construction des filtres et méthode de filtrage.

3.2. Compression de l'image par codage.

3.3. Primitives de niveau supérieur.

3.4. Organisation de la mémoire.

3.5. Description des programmes et résultats.

C H A P I T R E III

Opérateurs de classement des primitives

§1. UTILISATION DE LA NOTION DE DISTANCE POUR REGROUPER DES ELEMENTS COMPARABLES

1.1. Définition et rappels.

1.2. La notion de distance entre éléments comparables.

1.3. Les procédés d'agglutination ou "clustering".

1.4. Les ordonnances.

1.5. Les hiérarchies :

- hiérarchies indicées
- indices de similarité
- ultramétrique sous-dominante

§2. UNE METHODE DE REGROUPEMENT DEDUITE DES CONSIDERATIONS PRECEDENTES

2.1. L'algorithme.

2.2. Exemples.

2.3. Comment réaliser la première partie de l'algorithme.

2.4. Le choix des critères pour la définition d'une métrique.

2.5. Les résultats.

C H A P I T R E IV

Analyse syntaxique des formes - Opérateurs linguistiques

§1. FORMALISATION DES OPERATIONS DE RECONNAISSANCE DES FORMES
1.1. Introduction.
1.2. Opérateurs linguistiques.

§2. UN OPERATEUR INDEPENDANT DES TRANSFORMATIONS GEOMETRIQUES SIMPLES ET DES PETITES DEFORMATIONS.
2.1. Définition.
2.2. Le parenthésage et le principe d'affectation.
2.3. Méthode de comparaison de deux formes.
2.4. Description des expériences et résultats.

§3. AUTRES TYPES D'OPERATEURS LINGUISTIQUES
3.1. Nécessité de s'affranchir de certaines contraintes.
3.2. Plan général d'étude des opérateurs utilisés.

§4. UN OPERATEUR INVARIANT POUR LES ISOMETRIES
4.1. Définition de l'opérateur &(X).
4.2. Propriétés de &(X).
4.3. Algorithme de "découpage" du plan.
4.4. Reconstitution d'une forme X à partir de sa représentation U_X.
4.5. Méthode de comparaison de deux formes.
4.6. Résultats expérimentaux.

§5. UN OPERATEUR $\mathcal{O}(X)$ INVARIANT DANS LES PROJECTIONS PLANES

5.1. Définition de $\mathcal{O}(X)$.

5.2. Propriétés géométriques.

Enveloppe convexe.

5.3. Reconstruction de l'image à partir de U_X.

5.4. Comparaison de deux formes ayant le même nombre d'éléments.

5.5. Comparaison de deux formes ayant un nombre différent d'éléments.

5.6. Résultats.

PREAMBULE

L'objet général de cette étude est le traitement et la reconnaissance de données obtenues par des capteurs électromagnétiques à savoir les "Radars", à l'aide des systèmes généraux de manipulation d'information que sont les "ordinateurs".

La "Reconnaissance des Formes" est un ensemble de problèmes qui a préoccupé maints chercheurs, philosophes ou scientifiques et il nous semble en fait inabordable dans sa généralité. Nous croyons pouvoir y apporter une contribution en le restreignant au sujet choisi.

En 1967, nous n'avions aucune solution valable pour résoudre ce type de problème :

Un Radar délivre un signal électromagnétique. Habituellement, il est exploité par l'homme sous forme d'une image dite "PPI", fournie par un écran cathodique.

a. Comment fallait-il numériser les informations ainsi fournies pour les entrer dans une mémoire physique ?

La carte d'une région permet de prévoir dans une assez bonne mesure la situation telle qu'un observateur peut la voir sur une image PPI à l'aide d'un radar. Grâce à une carte en relief éclairée et photographiée ; on pourrait obtenir des "cartes prévisionnelles" à peu près satisfaisantes.

b. Comment obtenir des "cartes prévisionnelles", à partir d'une carte réelle, en utilisant un système informatique ?

A partir d'une carte prévisionnelle", l'observateur humain est capable de faire une comparaison avec l'image réelle PPI.

c. Comment effectuer ces comparaisons avec un système informatique qui utilis les solutions aux questions a. et b ?

En 1967, les seuls travaux connus portaient sur la reconnaissance de caractères imprimés, de chromosomes et quelques travaux théoriques.

Les trois problèmes posés ne sont pas indépendants. La phase de comparaison dépend de la nature des informations à comparer donc de la manière dont elles sont gardées en mémoire. C'est donc le type de représentation des images PPI et des cartes radars qui se posent dès le départ.

Ces images et cartes se présentent à l'observateur humain sous forme de taches noires sur fond blanc, ombres portées relief éclairé par le rayonnement électromagnétique. Or, pour une position données, ces taches varient en fonction de l'altitude d'observation. Ce qui explique l'échec des tentatives de comparer les taches entre elles.

Nous avions mis au point des procédés de traitement de taches connexes : extraction des contours, squelettes, [9] représentation par des sommes de formes simples etc...

Ceci nous a permis de dégager une idée essentielle qui a beaucoup simplifié le problème :

Les crêtes du relief sont des formes invariantes en fonction de l'altitude et même de la position d'observation au premier ordre près.

Dans ces conditions :

Le problème (a) consiste à savoir comment extraire d'une donnée radar les lignes de crête d'un relief et à les enregistrer dans les mémoires d'un système informatique.

Le problème (b) consiste à faire de même à partir d'une carte.

Le problème (c) est de faire les comparaisons entre les éléments enregistrés en mémoire dans les phases précédentes.

Un sous-produit inattendu de cette étude est de fournir une représentation nouvelle d'une "situation radar", avec l'avantage d'une compression considérable du volume d'information tout en étant précis et caractéristique d'une situation donnée.

Le chapitre I expose les procédés de préfiltrage et de prise d'information ; nous avons fait partie de l'équipe qui à l'Institut de Programmation a dégégé les idées et algorithmes AUBINEAU, HOUDAS, HORVAT, GUIHO.

Le chapitre II donne les algorithmes permettant d'extraire d'une image numérisée les lignes de crête cherchées. Cette méthode est originale. En effet, elle était présentée au Congrès d'Intelligence Artificielle de Rome, la communication étant inclue dans les préprints du Congrès. M. HOLDERMANN présentait à ce même congrès une communication orale très voisine et convenait que nous l'antériorisions.

Le chapitre III montre l'utilisation des procédés de classification dégagés par ailleurs, notamment en France par MM. BENZECRI, LERMAN, ROUX, DIDAY. Ils permettent d'établir un catalogue hiérarchisé de lignes de crêtes. Son utilité est de ranger une ligne quelconque dans une classe déterminée et de diminuer le nombre de comparaisons à effectuer.

Le chapitre IV présente les algorithmes de comparaison entre situations complexes de lignes de crête. Nous avons dégagé à cet effet, un type original d'opérateur invariant à certaines transformations géométriques. Ces opérateur sont de nature linguistique, bien que radicalement différents des grammaires de Chomsky. Ils peuvent être généralisés à bien d'autres problèmes et nous paraissent de grand intérêt en Reconnaissance des Formes.

Bien entendu, toutes ces méthodes ont donné lieu à des programmes qui se trouvent intégrés dans un software assez complexe, qui pourra être directement utilisé dans les applications.

Les applications en question peuvent permettre des réalisations pratiques comme une aide à la navigation aérienne et peut être son automatisation complète.

CHAPITRE I

Formalisation et langage adaptés aux processus de traitement des signaux et images

§1 LE PROBLEME DE LA RECONNAISSANCE DES FORMES PAR UN ORDINATEUR

1.1. Quelques définitions

Une représentation : grâce à nos sens nous percevons et reconnaissons des objets concrets. On sait noter le résultat de ces opérations de reconnaissance : une suite de sons, un mot ou un dessin, qui constituent à leur tour de nouveaux objets concrets (perceptibles par les sens).

Un signal : c'est une représentation du résultat d'une opération de mesure dépendant d'un paramètre, en général le temps.

Une image : représentation dépendant de deux paramètres d'espace.

Dans les deux cas, on utilise le nom de forme.
Ces mesures se traduisent concrètement par des ensembles de valeurs numériques que l'ordinateur utilisera.
Doit-on garder la totalité des informations fournies par le capteur, ou au contraire diminuer le volume des informations enregistrées ? La réponse sera donnée plus loin : elle dépend de l'objectif final. Par exemple, veut-on restituer l'image ou le signal reçu ou veut-on les identifier ?

1.2. Représentation des formes dans un ordinateur

Une "forme" est conservée en mémoire grâce à des échantillons numérisés ou quantifiés.

Si les physiciens ont choisi une représentation du type d'un objet mathématique s(x) ou S(x, y), c'est dans le but d'y associer des propriétés physiques telles que continuité, bornes de dérivation,... etc. Malheureusement, ces représentations impliquent des quantités d'information infinies et ne peuvent donc être transposées telles quelles sur ordinateur.

Il est bien connu que la limitation du volume de mémoire, donc l'échantillonnage discret, ne permet pas de représenter exactement les objets mathématiques choisis par les physiciens.

Il est certain d'autre part que les appareils de mesure ne fournissent que des échantillons discrets ; c'est seulement pour la commodité du raisonnement et des manipulations telles que les transformations géométriques que les représentations continues ont été adoptées.

Si nous voulons passer par l'intermédiaire de ces représentations continues, il nous faut les déduire des valeurs discrètes des mesures, puis à partir d'une représentation continue satisfaisante au point de vue mathématique et physique, en déduire des échantillons numériques, seuls acceptables par un système informatique.

Il semble que nous devrions nous passer de cet intermédiaire malgré son importance historique et enregistrer plutôt directement les échantillons mesurés.

Cependant les échantillons ainsi enregistrés vont nous permettre de procéder à des traitements tels que transformations géométriques, restitution du signal, reconnaissance ; et le problème se pose donc d'adapter cette représentation inhabituelle d'un phénomène physique à ces traitements.

C'est là où la représentation continue peut nous guider. Par exemple, le fait que les transformées de Fourier de s(x) sont à support borné, permet de choisir les paramètres d'échantillonnage de façon optimale.

Finalement, à partir des résultats de mesure d'appareils tels que capteurs optiques, écran cathodique ou Radar..., un certain nombre d'échantillons sont placés en mémoire. A précision donnée, le volume occupé de mémoire est réduit au minimum. C'est sur ces échantillons que seront réalisées les différentes

opérations de traitement et de reconnaissance. Ces échantillons constituent le premier niveau de représentation ou de reconnaissance d'une forme.

1.3. Les différentes étapes de la reconnaissance d'une forme

Les "analyseurs d'image" actuels sont capables de nous donner l'intensité lumineuse en un point d'une image. Comme nous l'avons dit précédemment, ce sont ces échantillons qui constituent le premier niveau de reconnaissance. La reconnaissance globale de l'image étant par exemple l'identification d'un véhicule terrestre, ou encore l'identification d'une voiture de type donné, ou mieux l'identification de la voiture Peugeot blanche portant tel numéro d'immatriculation... Nous pouvons penser qu'il n'est possible de passer directement de l'information brute remise en mémoire à l'information totale contenue dans l'image qu'en effectuant plusieurs opérations intermédiaires, et que ces opérations intermédiaires sont d'autant plus nombreuses que l'image à reconnaître est plus complexe.

Nous dirons que la reconnaissance d'une forme passe par différentes étapes correspondant à l'identification d'éléments de complexité variable. Ce qui est vrai pour la reconnaissance des images, l'est aussi chaque fois que nous voulons passer du stade de la prise de connaissance à celui de la compréhension et de l'utilisation de cette connaissance. Les analogies sont nombreuses dans les domaines aussi divers que les Sciences Naturelles, la théorie des jeux, la documentation automatique, la démonstration automatique de théorèmes, la programmation... Chaque fois les connaissances sont organisées (nous verrons plus loin qu'il s'agit de hiérarchisation des connaissances) de manière à permettre des rapprochements ou des déductions. Sans aller jusqu'à dire que toute opération d'intelligence relève d'un tel processus nous admettons que c'est très souvent le cas. Par exemple, les méthodes de documentation automatique utilisent des thésaurus construits de manière à éviter de consulter la totalité du dictionnaire lorsqu'on veut retrouver un mot. En Sciences Naturelles, on reconnaîtra un chien avant de dire que c'est un mammifère, mais aussi avant d'affirmer que c'est un Setter Irlandais ou un Berger Allemand...

Autrement dit, la Reconnaissance peut être plus ou moins facile suivant le niveau dans la classification que l'on considère et on remonte ou l'on descend dans l'arbre symbolisant cette classification selon que l'on veut rattacher l'objet à identifier à une classe plus vaste ou lui trouver des propriétés spécifiques.

D'une manière générale, une opération de reconnaissance de forme est inséparable d'un opérateur concret.

Supposons qu'une image soit représentée par 10^6 échantillons à deux niveaux (0 et 1). Soit l'opérateur additionnant la valeur arithmétique des échantillons et décidant si la somme est paire ou impaire. Cette opération sépare toutes les représentations en deux classes. Par contre, on peut se demander si cette représentation a une signification.

Nous cherchons donc des opérateurs dont le résultat nous rapproche de nos propres "opérateurs" de reconnaissance. Même si nous n'y aboutissons pas du premier coup, nous essayons de déterminer des opérateurs dont les résultats sont interprétables de façon significative. on considère par exemple que les lignes, "segments", intersections etc... d'une image constituent des éléments caractéristiques.

Nous "savons" ce qu'est un segment en examinant une image mais comment le définir à la machine ? Nous ne pouvons le faire qu'en décrivant l'algorithme ad hoc (opérateur algorithmique).

Tout opérateur utilisable sur ordinateur se traduit en fait par un programme qui à partir de données fournit des résultats, sous forme concrète, des états de registre. C'est l'interprétation sémantique de ces états de registre qui nous sert en première analyse à caractériser un opérateur.

Opérateurs numériques ou paramétriques

Le résultat final est interprété comme une valeur numérique. Etant donné que les échantillons de mesure eux-mêmes représentent des amplitudes ou intensités, ce type d'opérateur est le plus souvent utilisé à ce premier niveau de reconnaissance.(Cf. plus loin détection des segments dans des images de ligne).

Opérateurs de classification

Ces opérateurs permettent de regrouper en paquet ou nuages (clusters) les éléments comparables d'une classe de représentations. Par exemple, des éléments de mesure multidimensionnels en médecine, botanique, zoologie... Ces opérateurs utilisent des mesures de distance entre éléments deux à deux. Le résultat consiste à affecter une étiquette commune aux éléments d'un même nuage. Il n'est donc plus numérique.

Opérateurs linguistiques

Ces opérateurs sont bien connus en compilation, c'est-à-dire dans l'analyse des langages de programmation. Ils utilisent des grammaires ou règles de récritures portant sur les mots d'un langage. Une règle de récriture est donc un opérateur dont le résultat est un mot et qui permet de décider si telle ou telle chaîne de caractères fait partie d'un langage. Elle permet de plus de reconnaître le mot et de le traduire.

Certains auteurs ont proposé d'étendre ces procédés aux opérations d'analyse des images. Les résultats obtenus ne sont pas probants.

Nous insisterons sur la nécessité de trouver des opérateurs invariants en fonction des transformations habituelles sur les images : déplacements, homothéties, rotations, ainsi que certaines déformations.

Nous avons ainsi mis en évidence des opérateurs tout-à-fait nouveaux, et qui, curieusement, se rangent dans cette classe d'opérateurs linguistiques : ils utilisent les "noms" des formes primitives d'un niveau déterminé et donnent pour résultats des "mots" construits sur l'alphabet des noms. Ces opérateurs

méritent donc bien le qualificatif de linguistique ; cependant ils ne ressemblent ni de près ni de loin aux opérateurs des grammaires génératrices de Chomsky.

Leur mise à jour, l'étude de leurs propriétés, leur utilisation pour des opérations puissantes de reconnaissance sont probablement la partie la plus originale de nos travaux. Nous les développons au chapitre IV.

§ 2. PRISE ET CONSERVATION DE L'INFORMATION - ECHANTILLONNAGE

2.1. L'information d'une image

Posons la question du passage de la représentation sous forme d'échantillons numériques à la représentation continue.

Les physiciens utilisent une fonction continue S(x, y) pour représenter l'éclairement en un point (x, y) du plan. Cette fonction S(x, y) a pour Transformée de Fourier (T. F.) une fonction à support borné. La conséquence de cette propriété permet l'application du théorème d'échantillonnage de NYQUIST-SHANNON ; rappelons-le pour une fonction à une dimension s(x) : il est possible, suivant certaines conditions sur α et g(x), de représenter exactement s(x) par une somme infinie de termes :

$$S(x) = \sum_{-\infty}^{+\infty} s(p\alpha)\, g\left(\frac{x}{\alpha} - p\right) \qquad (2.1)$$

où p est un entier et α est l'intervalle d'échantillonnage.
Cette représentation est encore inutilisable dans un ordinateur, puisqu'en principe il faudrait conserver un nombre infini d'échantillons de précision infinie.

Remarque 1 : la formule (2.1) donne la valeur exacte de s(x) pour x appartenant à l'intervalle $[p\alpha, (p + 1)\alpha]$. On a l'habitude de prendre, pour g(x), la fonction de NYQUIST-SHANNON

$$g_{N-S}(x) = \frac{\text{Sin } \pi x}{\pi x}$$

mais d'autre fonctions peuvent être utilisées (cf.....)
Si a est la borne supérieure du spectre de s(x) :

$$\alpha \leqslant \frac{1}{2a}$$

La fonction $g_{NS}(x)$ est celle pour laquelle cette borne peut être atteinte. Toutes ces fonctions sont décroissantes à partir de l'origine mais $g_{NS}(x)$ décroît le plus lentement.

Remarque 2 : les échantillons entourant x interviennent d'autant moins dans la somme donnant s(x) qu'ils sont plus éloignés de x. Ainsi plus g(x) décroît rapidement moins il faut d'échantillons pour obtenir s(x) avec une précision donnée.

2.2. Utilisation d'un support borné

Nous avons recherché[1] des fonctions $g_n(x)$ satisfaisant aux conditions suivantes :

- fonctions symétriques
- support borné de mesure n + 1
- fonctions polynomiales de degré n

Alors si

$$s_n(x) = \sum_p s(p\alpha) g_n\left(\frac{x}{\alpha} - p\right) \tag{2.2}$$

$$|s(x) - s_n(x)| \leqslant \epsilon_n \tag{2.3}$$

avec

$$\epsilon_n \leqslant (\pi\alpha a)^n \sqrt{\frac{2}{\pi n}}\, M \tag{2.4}$$

M est la borne de $|s(x)|$

On constate sur (2.4) que $\epsilon_n \to 0$ si $\alpha \leqslant \frac{1}{\pi a} < \frac{1}{2a}$

Naturellement l'erreur ϵ_n sera d'autant plus petite que n sera grand.

La figure 1.2 représente les premières fonctions g_n : (g_0, g_1, g_2).

A la limite, si $n \to \infty$, on retrouve $g_{ns}(x)$

(1) Cf. les travaux de Messieurs AUBINEAU, HOUDAS et HORVAT [2] et [3].

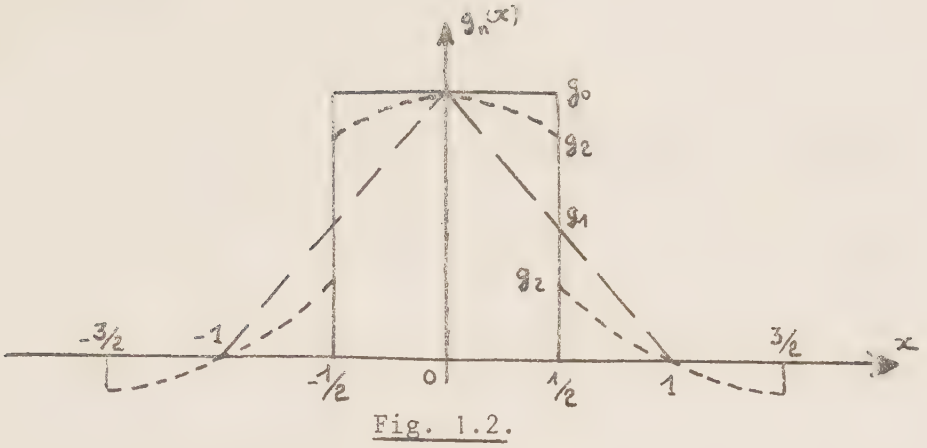


Fig. 1.2.

2.3. Utilisation d'échantillons discrets

La formule (2.2) fournit les valeurs exactes de s(px). Supposons que nous n'ayons que des valeurs discrètes, au total N2, et toutes équidistantes. Si e(pα) est la valeur de l'échantillon quantifié on peut écrire que

$$|e(p\alpha) - s(p\alpha)| \leq \epsilon_d$$

Et si

$$|s(x)| \leq M \quad \text{alors } \epsilon_d N_2 = M$$

Pour représenter s(x) on pourra donc prendre

$$\sigma_n(x) = \sum_p e(p\alpha) g_n(\frac{x}{\alpha} - p)$$

et on démontre que

$$|\sigma_n(x) - s(x)| \leq \epsilon_n + \epsilon_d = \epsilon$$

$\epsilon_n \rightarrow$ erreur due au procédé g_n

$\epsilon_d \rightarrow$ erreur due à la discrétisation

Il s'agit alors de faire un choix optimum de α et de n.

Il faut pour cela que ϵ soit peu supérieure à ϵ_d.

Sur ordinateur, ces méthodes ont un intérêt évident : pour une précision donnée, le nombre n d'échantillons est fixé. Les mêmes propriétés ont été démontrées par M. GUIHO pour des signaux à deux dimensions (cf [13]).

Résultat important : si le support minimal est un rectangle, alors
$g(x, y) = g(x)g(y)$
Par exemple : si $\epsilon \sim 10^{-2}$ l'emploi de g_3 au lieu de g_0 permet de multiplier α par 100.
Si $\epsilon \sim 10^{-4}$ l'emploi de g_4 au lieu de g_1 multiplie aussi α par 100.
Dans le cas d'une surface, la quantité d'information à conserver en mémoire est donc divisée par 10^4 !

En conclusion, un procédé pratique à utiliser sur ordinateur a été mis au point, à partir des qualités prévisibles de $s(x)$ ou $S(x, y)$ il permet de les reconstituer à une précision ϵ donnée a priori en utilisant les échantillons numériques gardés en mémoire.

De plus ϵ étant donné, il est possible de minimiser le volume de mémoire à conserver.

Les conséquences pratiques sont en fait tout à fait importantes pour les expérimentateurs qui utilisent le plus souvent au hasard leur précision d'échantillonnage.

Avant même de procéder à des opérations de reconnaissance, il importe d'optimiser la prise de l'information sous peine soit d'être incapable de retrouver les informations utiles, soit de saturer complètement mémoires et temp de calcul.

2.4. Application à la représentation des "lignes"

La "ligne" est une des formes les plus élémentaires que l'on puisse rencontrer. Sa représentation pose pourtant des problèmes qu'il nous a été difficile de résoudre.

En effet, nous avons l'habitude de considérer l'objet abstrait ligne en géométrie (où nous définissons un cercle, une droite ou plus généralement la limite entre deux domaines du plan comme un ensemble de points de R^2. Pourtant nous représentons une ligne au tableau ou sur une feuille de papier par une figure concrète qui est une grossière approximation de la définition mais qui ne nous gêne nullement pour reconnaître un cercle, une droite...

De même, dans la mémoire d'un ordinateur, une ligne est représentée par un ensemble de points correspondant à une figure concrète et non à la définition théorique de la ligne. Le choix en est guidé par les possibilités de restitution et de transformation d'une représentation acceptable de ligne à partir d'échantillons.

CHAPITRE II

Traitement des images

§1 TRANSFORMATIONS SUR UNE REPRESENTATION

Soit la fonction continue S(x,y) représentation habituelle d'une image. Une transformation physique de cette image donne un résultat S'(x', y'). On peut parler d'une application dont l'objet est S(x, y) et l'image S'(x', y'). Nous allons considérer des applications correspondant à des opérations concrètes. Nous nous limiterons à :

- des transformations ponctuelles
- des transformations linéaires

1.1. Les transformations ponctuelles

Ces transformations sont définies par le fait qu'à un point M(x, y, S) correspond un moint M'(x', y', S') tel que :

$$\begin{cases} x' = h'(x, y) \\ y' = k'(x, y) \end{cases} \quad \text{ou} \quad \begin{cases} x = h(x', y') \\ y = k(x', y') \end{cases} \qquad (2.1)$$

Sur le plan mathématique, la manipulation de ces équations ne présente aucune difficulté. On pourra par exemple, si S(x, y) possède des dérivées d'ordre n en déduire celles de S'(x', y').

Sur le plan concret, il en est autrement. On ne peut plus supposer que M et M' appartiennent à $\mathbb{R}^3$ et les relations (2.1) ne peuvent être qu'approchées. Dans l'ordinateur, une relation fonctionnelle entre M et M' n'est représentable qu'à une erreur ϵ près.

La transformation inverse ne peut donc pas redonner M à partir de M'. Ces notions ont été précisées par l'étude qu'en a faite G. GUIHO [4]. Supposons que S(x, y) soit représentable par une somme $\sigma_n(x, y)$ utilisant les fonctions polynomiales $g_n(x, y)$ à support borné :

$$\sigma_n(x, y) = \sum_{p, q} e(p, q) g_n\left(\frac{x}{\alpha} - p, \frac{y}{\beta} - q\right) \tag{2.2}$$

α et β sont les intervalles d'échantillonnage respectivement en x et y ; p et q sont des entiers : n est le degré des polynomes $g_n(x, y)$ ainsi que le côté du carré support de g_n.

L'erreur de discrétisation ϵ_d est donnée par :

$$|e(p\alpha, q\beta) - S(p\alpha, q\beta)| \leqslant \epsilon_d \tag{2.3}$$

Alors

$$|S - \sigma_n| \leqslant \eta_n \epsilon_d + \epsilon_n = \epsilon \tag{2.4}$$

en général $\eta_n \# 1$

Dans le cas d'un déplacement, une rotation, plus généralement d'une isométrie, les échantillons de S'(x', y') s'obtiennent à partir de (2.2) et l'erreur sur la nouvelle représentation est :

$$\epsilon' = \eta_n \epsilon + \epsilon_n \# \epsilon_d + 2\epsilon_n \tag{2.5}$$

Une transformation ponctuelle entraîne donc une erreur supplémentaire de l'ordre de ϵ_n et une suite de p transformations aboutit à une erreur totale :

$$\epsilon_p \# \epsilon_d + \varphi(p)\, \epsilon_n \tag{2.6}$$

Il est probable que $\varphi(p) \sim \sqrt{p}$ à cause du caractère aléatoire du signe des erreurs successives.

Quoiqu'il en soit, les transformations envisagées ne peuvent qu'augmenter l'imprécision initiale.

L'explication de certains paradoxes des représentations échantillonnées devient alors claire : afin de réduire le volume de mémoire, n'utiliser que deux valeurs discrètes (0, 1) d'échantillonnage est tentant mais l'erreur est alors de l'ordre de la moitié du maximum du module ! Supposons par exemple une transformation Θ appliquée à des échantillons 1 représentant une surface connexe.

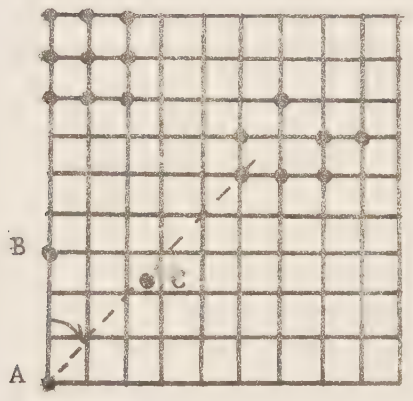


Figure 2.2

La figure 2.2. est un exemple de représentation d'une surface par des échantillons discrets : un échantillon 1 est matérialisé par un point à une intersection du maillage. L'absence de point caractérise les échantillons 0. Ainsi, les neufs points les plus à gauche peuvent représenter une surface symétrique qui pouvait être, avant discrétisation, un carré, un polynome régulier ou même un cercle.

Essayons d'effectuer une transformation géométrique simple sur une telle figure, en restant dans le maillage donné. Une rotation R(A, α) de centre A et d'angle α, transforme le point B en un point C qui ne sera pas sur le maillage (sauf pour $\alpha = k\frac{\pi}{2}$). On peut alors choisir de prendre le point du maillage le plus proche de C, soit C1, ou encore de prendre le point C2 dont les coordonnées sont les parties entières de celles de C. Dans tous les cas, la transformation inverse R^{-1} ne permet pas de retrouver la figure de départ. En particulier, un certain nombre d'échantillons disparaissent, comme sur l'exemple de la figure 2.2 où la surface représentée par 9 échantillons, ne comporte plus que 7 échantillons après une rotation.

Comme nous l'avons souligné précédemment, la représentation concrète d'une droite, d'un cercle ne correspond pas à la définition abstraite droite ou cercle. Reprenons l'exemple de la représentation de "lignes" noires sur fond blanc.

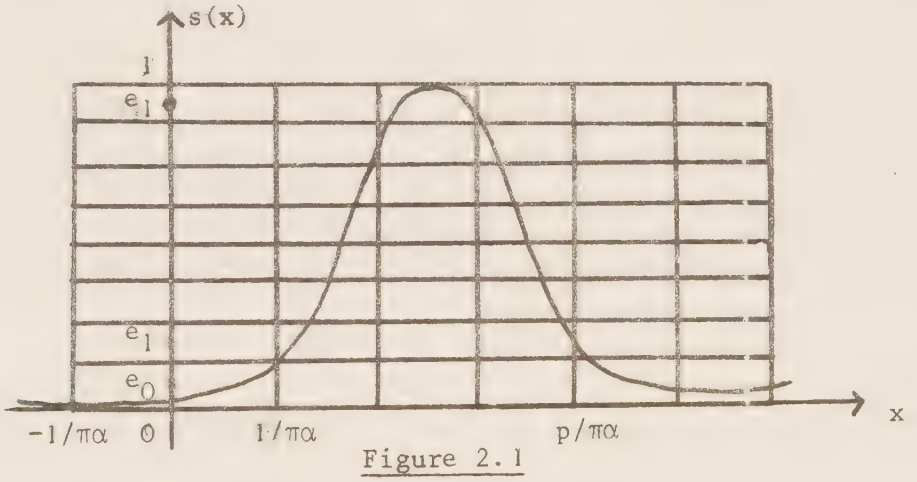


Figure 2.1

Une coupe transversale de l'image d'une ligne peut se représenter par s(x) qui a l'allure de la figure 2.1.
Si S(u) est la transformée de Fourier (à support borné) de s(x)

$$|S(u)| = 0 \qquad \text{si } |u| \geqslant a$$

Le théorème de BERNSTEIN (cf. ARSAC [ ]) indique alors que

$$|s'(x)| \leqslant \pi a \qquad |s''(x)| \leqslant \pi a^2 \tag{2.7}$$

ce qui exclue les fronts raides ou les courbures nulles...

En supposant que s(x) varie entre 0 et 1, et si e_0, e_1,..., e_7 sont huit échantillons possibles, il est facile de voir, grâce à (2.7), qu'au niveau 1/2 la largeur est au minimum $2/\pi a$. Elle est au moins $3/\pi a$ au niveau 1/8.

Pour utiliser un "procédé d'interpolation g_n" sans erreur excessive, l'intervalle d'échantillonnage α doit être inférieur ou égal à $1/\pi a$ donc il faut prendre au moins 3 échantillons différents de 0.

En prenant 4 échantillons successifs différents de 0, l'erreur serait de l'ordre de 1/16.

D'après ce que nous avons vu (formule 2.5) une transformation portant sur les échantillons eux-mêmes entraîne une erreur de l'ordre de 1/8 sur la représentation de s(x), ce qui la rend encore très acceptable. Le "paradoxe" a disparu...

1.2. Les transformations linéaires

Rappelons brièvement le formalisme de l'"Analyse Harmonique". Si l'on désigne par s l'ensemble des représentations "objets" et par r l'ensemble des représentations "images", qui sont tous deux des espaces vectoriels, en général sur le corps des réels. L'application T qui fait correspondre r à s est linéaire si :

$$T : s \to r \text{ entraîne } T : (\lambda_1 s_1 + \lambda_2 s_2) \to (\lambda_1 r_1 + \lambda_2 r_2)$$

Dans le cas de "signaux" électroniques ou optiques s et r appartiennent à ^{2} ou ^{3}. Alors si T est telle que :

$$T : s(x - x_0) \to r(x - x_0) \tag{2.8}$$

Les fonctions harmoniques forment une base de fonctions propres, c'est-à-dire qu'elles restent inchangées dans la transformation T :

$$T : e^{iux} \to (u)\, e^{iux} \tag{2.9}$$

Ce sont ces deux conditions de linéarité et d'invariance aux translations qui font l'intérêt des représentations par transformée de Fourier ; en effet si :

$$s(x) = \int_{-\infty}^{+\infty} S(u)\ e^{iux}\ d_u \qquad (2.10)$$

La réponse ou image de s(x) est donnée par :

$$r(x) = \int_{-\infty}^{+\infty} \gamma(u)\ S(u)\ e^{iux}\ d_u \qquad (2.11)$$

Naturellement, il faut pouvoir adapter un tel processus aux possibilités de l'ordinateur. C'est pour cela que l'algorithme "F. F. T." (Fast Fourier Transform) a été mis au point. Si on représente s(x) à l'aide d'une somme de translatées, la T. F. se calcule comme une somme finie et l'algorithme "F. F. T." permet d'optimiser le calcul de cette somme.

Malheureusement, aucun calcul d'erreur n'accompagne l'adaptation F. F. T. de la transformée de Fourier sur ordinateur. Les erreurs ont pour source principale la discrétisation de départ des échantillons. Les procédés de filtrage en spectre que nous allons évoquer impliquent une double F. F. T. En principe les algorithmes proposés sont parfaits si la précision d'échantillonnage est infinie, en pratique de graves erreurs s'introduisent sur l'ensemble de l'image. C'est la raison pour laquelle des procédés de traitement direct ont été mis au point. Ils sont plus rapides et les erreurs restent localisées.

1.3. Les filtres digitaux

Soit s(x) la représentation d'un signal à une dimension, r(x) sa transformée par un filtrage linéaire. Soit e(p) l'échantillon discrétisé de s(x).

$$|e(p) - s(x)| \leqslant \epsilon_d$$

Il est possible d'écrire (cf. ( )) :

$$\sigma_n(x) = \sum_p e(p)\ g_n\ (x - p) \qquad (2.12)$$

Avec

$$|\sigma_n(x) - s(x)| \leqslant \eta_n \epsilon_d + \epsilon_n = \epsilon$$

Si nous connaissons la transformée de Fourier $G_n(u)$ de $g_n(x)$ et la transmission spectrale $\gamma(u)$ du filtre , la réponse $\rho_n(x)$ de ce dernier à $g_n(x)$ est donnée par :

$$\rho_n(x) \overset{T.}{\rightarrow} \overset{F.}{} R_n(u) = \gamma(u) \cdot G_n(u) \qquad (2.13)$$

La réponse $\omega_n(x)$ de $\mathcal{F}$ à $\sigma_n(x)$, c'est-à-dire la réponse approchée de $\mathcal{F}$ au signal $s(x)$ peut se déduire par :

$$\omega(x) = \sum_p e(p)\ \rho_n\ (x - p) \qquad (2.14)$$

En supposant que ρ_n puisse se représenter par une formule (2.12) on pourra écrire :

$$\rho_n(x) \# \sum_q \rho_d(q)\ g_n\ (x - q) \qquad (2.15)$$

où ρ_d est l'échantillon discrétisé de ρ_n.
Par suite

$$\omega_n(x) \# \sum_i \omega(i)\ g_n\ (x - i) \qquad (2.16)$$

avec

$$\omega(i) = \sum_p e(p)\ \rho_d\ (i - p) \qquad (2.17)$$

ou

$$\omega(i) = \sum_q e(i - q)\ \rho_d\ (q) \qquad (2.18)$$

Ces deux dernières formules sont celles des <u>filtres digitaux</u> proposés par KUO et KAISER [6], chapitre "Digital Filters".

Remarquons entre autre, que le formalisme de la formule (2.17) ou (2.18) a été proposé directement pour les réseaux de neurones et qu'il a inspiré les "Perceptrons".

Equivalence entre filtres linéaires et filtres digitaux

Supposons que $\gamma(u)$ soit à support borné :

$$|\gamma(u)| = 0 \quad \text{si} \quad |u| \geqslant \frac{1}{2}$$

alors, d'après (2.13)

$$|R_n(u)| = 0 \quad \text{si} \quad |u| \geqslant \frac{1}{2}$$

Le spectre de $\rho_n(q)$ que nous appellerons $\widetilde{R}_n(u)$ s'obtient en rendant périodique de période 1 la fonction R_n.

Si on désigne par G_∞ la limite de G_n pour $n \to \infty$, d'après les hypothèses faites :

$$R_n = \widetilde{R}_n \cdot G_\infty \tag{2.19}$$

$$G_\infty(u) = 1 \quad \text{si} \quad |u| \leqslant \frac{1}{2}$$
$$= 0 \quad \text{si} \quad |u| > \frac{1}{2}$$

(2.19) est la relation d'échantillonnage de NYGUIST-SHANNON ; on la reconnaît, en repassant à l'espace des signaux :

$$\rho_n = \widetilde{\rho} \star g_\infty \tag{2.20}$$

où $\widetilde{\rho}_n$ est la suite des échantillons et $\star$ le signe de la convolution. Il est facile de déduire que

$$\gamma \cdot G_m = R_n \cdot G_\infty \tag{2.21}$$

Par hypothèse $|\gamma| = 0$ si $|u| > \frac{1}{2}$ et la relation (2.21) ne présente de l'intérêt que dans l'intervalle $(-\frac{1}{2}, +\frac{1}{2})$. Alors :

$$\gamma = \frac{\widetilde{R}_n}{G_n} \tag{2.22}$$

Cette dernière relation n'est qu'approchée à cause de la discrétisation. De plus les valeurs de n utilisées sont telles que G_n est voisin de l'unité sauf peut être aux extrémités de l'intervalle où R_n est voisin de zéro. Il est donc possible d'écrire :

$$\gamma \# \tilde{R}_n \quad \text{si} \quad |u| \leqslant \frac{1}{2} \tag{2.23}$$

On peut obtenir $\tilde{R}_n(u)$ très facilement :

$$\tilde{R}_n(u) = a_0 + \sum_q a_q \cos 2\pi qu + b_q \sin 2\pi qu \tag{2.24}$$

Avec

$$a_0 = \rho(0)$$
$$2aq = \rho(q) + \rho(-q)$$
$$2bq = \rho(q) - \rho(-q)$$

$\tilde{R}_n(u)$ est donc un polygône trigonométrique dont l'ordre est égal à $\frac{m}{2}$ si m est pair, à $\frac{m-1}{2}$ si m est impair, m étant le nombre total de coefficients du filtre digital.

§2. LE PROBLEME DU RECALAGE D'UN AVION

2.1. Les données actuelles

La photo de la planche [II01] représente le "scope" du RADAR d'un avion qui survole une région montagneuse. Les tâches sombres représentent l'ombre portée de l'onde électromagnétique. Le front avant de ces tâches sombres correspond donc aux contours des obstacles rencontrés par le rayonnement, c'est-à-dire que ce sont approximativement les crêtes du relief.

Afin de pouvoir exploiter ce type d'information pour vérifier la position de l'avion à un instant donné, le pilote compare ce qu'il voit sur le scope à des photos représentant la région survolée. Pour faciliter cette comparaison, un procédé a été mis au point par l'Armée de l'Air : la région survolée est reconstituée à échelle réduite, en platre ou en carton. Le relief est éclairé par une lumière rasante et photographié à la verticale. On obtient ainsi une série de clichés repérant le "plan de vol" et le pilote a donc alors à sa disposition un catalogue qu'il consulte progressivement en comparant chaque photo à l'image qui apparaît sur le scope radar. Ce catalogue a été appelé "carte pseudo-radar" (voir planche [II02]).

Une petite variante à ce procédé permet d'éviter de changer de cliché trop souvent. Il consiste à photographier le relief en déplaçant progressivement l'appareil photographique, de manière à simuler le déplacement de l'avion. On obtient, pour un plan de vol déterminé, un cliché unique représentant par exemple une centaine de kilomètres d'une bande de 10 km de largeur. La photo obtenue est appelée "carte prévisionnelle" (voir planche [II03]).

Devant ce type de représentations, il semble que le pilote retienne plutôt la forme, l'importance et la disposition relatives des tâches sombres pour comparer deux situations. Nous avons consacré une longue période à l'étude des formes planes connexes, à leur simplification et leur codage pour faciliter leur reconnaissance par ordinateur. Nous avons ainsi été amenés à mettre au point des algorithmes d'extraction de contours (cf. C. R. A. S..... locomotive) ainsi que des méthodes pour déterminer le squelette de formes connexes (cf. article sur les chromosomes...).

Cependant, les résultats obtenus pour le problème de la reconnaissance des cartes radar nous ont conduit à la conclusion suivante :

Le type de représentation correspondant à ces "cartes radar" ne permet pas de s'affranchir de contraintes telles que la nécessité de comparer des images correspondant à la même altitude, le même direction etc... Il est apparu que les seuls éléments indépendants de l'altitude, de la direction de vol et de l'appareil de mesure, soient les crêtes du relief.

Ainsi, cette première étude a permis de dégager une nouvelle perspective pour les représentations des données Radar ; il est en effet possible d'envisager dès maintenant une modification assez simple du matériel pour réaliser par exemple la dérivation du signal vidéo et fournir directement ces éléments caractéristiques parcequ'invariants que sont les crêtes.

Evidemment, pendant la phase expérimentale, il n'est pas question de réaliser du matériel coûteux ou de modifier le matériel existant pour obtenir des données conformes à ce premier résultat. Nous avons donc travaillé en simulation à partir des photos qui nous ont été proposées. Dans les paragraphes qui suivent nous donnons les différentes étapes de cette simulation qui a permis de mettre au point des algorithmes de filtrage digital indispensables pour comprimer l'information d'une part et pour extraire des lignes d'autre part.

2.2. Application du filtrage linéaire pour réduire le volume de ces données

Le discrétiseur d'image "Calife"

L'appareil "Calife" du Laboratoire Central de l'Armement est capable de mesurer, par transparence, l'intensité lumineuse d'un point d'une diapositive éclairée par un "flying-spot".

La dimension du spot est d'environ 60 μ.

Les supports des images sont des diapositives 24 x 36 mn.

Pour un pas d'exploration égal au diamètre du spot, on obtient au maximum 512 échantillons par ligne.

La fenêtre d'exploration peut être modifiée par programme. Ainsi une diapositive 24 x 36 donne lieu à 512 x 350 mesures d'intensités lumineuses au maximum. Ces mesures sont codées par 10 bits et enregistrées sur bandes magnétiques. Finalement, une image telle que celle de la figure [ ] est

représentée par un tableau d'environ 18 000 mots correspondant à autant de valeurs P(k, l) de l'intensité lumineuse de l'image donnée aux points de coordonnées (k, l).

Cette information, trop détaillée pour l'usage qu'on doit en faire est lourde à manipuler telle quelle. Il convient de la réduire avant même de tenter des comparaisons, sans perdre trop d'informations intéressantes.

Compression de l'image discrétisée

Supposons que le plus petit détail à conserver ait une taille égale ou supérieure à n fois le pas de discrétisation, ce qui revient à dire que le plus petit détail conservé doit être décrit par au moins n échantillons. Nous prenons alors un filtre P'_{IJ} défini par :

$$P'_{IJ} = \sum_{i,\,j} \frac{a_{ij}\, P_{ij}}{N}$$

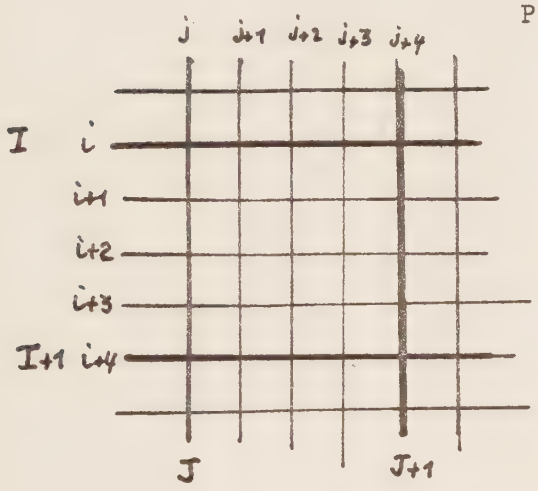


Fig. II.3

La somme en i et j se faisant sur un domaine qui sera par exemple le carré de côté n.
Soit pour

Etant donné la discussion faite sur la représentation d'une ligne, 4 échantillons consécutifs non lus correspondent au signal impulsionnel le plus bref.

Le résultat d'un tel filtrage est montré sur la planche[2]où on a pris N = 1 024, n = 4 et $a_{ij} = 1 \quad \forall_{i,\,j}$; pour pouvoir utiliser l'imprimante, le nombre de niveaux a été ramené à 8.

Le choix des $a_{ij} = 1$ n'est pas "le meilleur". C'est cependant le plus simple et il donne satisfaction avec les hypothèses retenues. En particulier, l'impression pour l'oeil de traits continus est toujours vraie après discrétisation, grâce à l'utilisation d'au moins trois échantillons dans le sens de l'épaisseur et de plusieurs niveaux d'intensité :...

Les filtres digitaux à supports bornés ont été introduits comme des opérateurs de transformation sur une représentation d'image.

Dans le prochain paragraphe, nous allons montrer leur utilisation comme opérateurs de reconnaissance de formes élémentaires de premier niveau. Il est incontestable que ce sont les opérateurs les plus simples à mettre en oeuvre sur un calculateur numérique ou analogique. Nous les appellerons opérateurs de convolution.

§3. EXTRACTION DE PRIMITIVES DANS LE CAS D'IMAGES FORMEES DE LIGNES

1.1. Extraction des caractéristiques ou primitives de niveau 1

Rappelons qu'un filtre linéaire est un opérateur de convolution tel que :

$$P'_\theta(I, J) = \sum_{i,\, j \in \bar{a}_\theta} a_\theta(i, j)\, P\,(I + i, J + j) \qquad (2.25)$$

où a_θ sont les coefficients du filtre.
$\bar{a}_\theta$ représente le support sur {i, j} de la fonction $a_\theta(i, j)$.
Les coefficients $a_\theta(i, j)$ sont déterminés expérimentalement (ou statistiquement) sur un certain nombre d'exemples des éléments à extraire. Nous avons choisi d'extraire des segments de ligne et pour cela nous avons imposé les conditions suivantes pour les filtres :

- en coupe, il doivent avoir une allure analogue à celle des éléments à extraire,
- leur longueur doit être nettement supérieure à la largeur d'une ligne.
- il faut autant de filtres que de directions envisagées.

Puisqu'une ligne possède en moyenne 3 échantillons dans son épaisseur nous avons pris des supports $\bar{a}_\theta$ avec 3 x 5 éléments ;
4 directions possibles dans l'intervalle $[0, \frac{\pi}{4}]$.

On obtient les autres filtres (directions entre $\frac{\pi}{4}$ et 2π) par symétrie. La longueur des filtres a été établie de manière à obtenir un poids presque constant et tel que la moyenne des longueurs comporte 5 échantillons.

La figure 2.2 présente les coefficients d'une série de filtres de ce type. Les paramètres d'un filtre sont :

- sa direction
- son origine
- son extrémité
- son pas de progression

On peut remarquer que chaque filtre possède deux points particuliers qu'on appelle ses "foyers" F1 et F2. Le premier, F1, permet de positionner le filtre au point origine de l'exploration. Le foyer F2 fournit la position du point de départ de l'exploration suivante.

D'où l'algorithme de filtrage

Une "ligne" est rencontrée lorsque, au cours du balayage, on trouve un point dont le niveau d'intensité S(x, y) est supérieur à un seuil S_0. Ce dernier est un paramètre déterminé expérimentalement pour un type d'image. Les différents filtres sont alors essayés successivement en plaçant leur foyer F1 en ce point. L'un d'entre eux donne un résultat (après convolution) plus grand que tous les autres. Si cette valeur est supérieure à un seuil S1, la direction correspondante est sélectionnée. Le foyer F2 devient le point de départ de l'exploration suivante. Si la plus grande valeur obtenue par convolution est inférieure au seuil S1, on reprend le balayage systématique. C'est le cas en particulier lorsqu'on a affaire à du "bruit", lorsqu'on est à l'extrémité d'une ligne, ou lorsqu'on est sur le "cadre" de l'image explorée.

Remarque : quand une direction est sélectionnée, les points correspondant au filtre sont effacés ; à l'exception du foyer F2 et des points immédiatement voisins. Ceci pour permettre le départ de l'exploration suivante et assurer la continuité des lignes.

Résultats pratiques de ce filtrage

Ils apparaissent sur la planche 3. Chaque segment schématise un des filtres utilisés donc représente une direction sélectionnée. Il y a eu élimination des points et des petites lignes isolées (qu'on peut appeler "bruit"). Evidemment, en faisant varier le paramètre S1, on pourra être plus ou moins sévère sur les détails à retenir.

Nour dirons que nous avons obtenu, par ce filtrage, des primitives de niveau 1 et nous les désignerons par $\pi 1$.

3.2. Nouvelle compression de l'image

Dans les exemples sue nous avons traités une image est représentée au départ par environ 18 000 mots (matrice des intensités). Le filtrage qui vient d'être décrit a réalisé l'extraction de l'ensemble des lignes qui constituaient l'image.

Etant donné que chaque ligne est composée d'un certain nombre de segments de longueur constante, on pourra décrire chaque ligne par une représentation très condensée et l'image sera réduite à la représentation de l'ensemble des lignes.

Par exemple, on donne, pour chaque ligne, les coordonnées (x, y) du point de départ, puis les codes des directions rencontrées successivement. Ainsi une ligne sera représentée par une chaîne de caractères.

Si on admet que, dans les images considérées, la longueur moyenne d'une ligne est de 10 segments élémentaires ; et si l'image comportait une vingtaine de lignes, l'occupation mémoire d'une telle image est alors réduite de 18 000 mots à 200 ou 300 mots.

3.3. Détermination de primitives de niveaux supérieurs

Le dernier mode de représentation proposé est encore assez redondant dans beaucoup de cas, puisque les parties rectilignes sont décrites par un grand nombre de segments. On peut donc décider d'associer les segments élémentaires en éléments plus importants lorsqu'ils sont jointifs et de même direction

Par exemple : la ligne ci-dessous

représentée par le code X Y 1 7 4 4 4 4 4 4

pourra devenir X Y 1 7 (6 4)

La concaténation des primitives π1 fournit des primitives de niveau 2 que nous appellerons π2. Cette concaténation se fait très facilement à partir des codes eux-mêmes.

3.4. Organisation en mémoire

Nous avons voulu que l'implantation en mémoire d'une image soit suffisamment dynamique pour permettre de l'explorer à différents niveaux de précision.

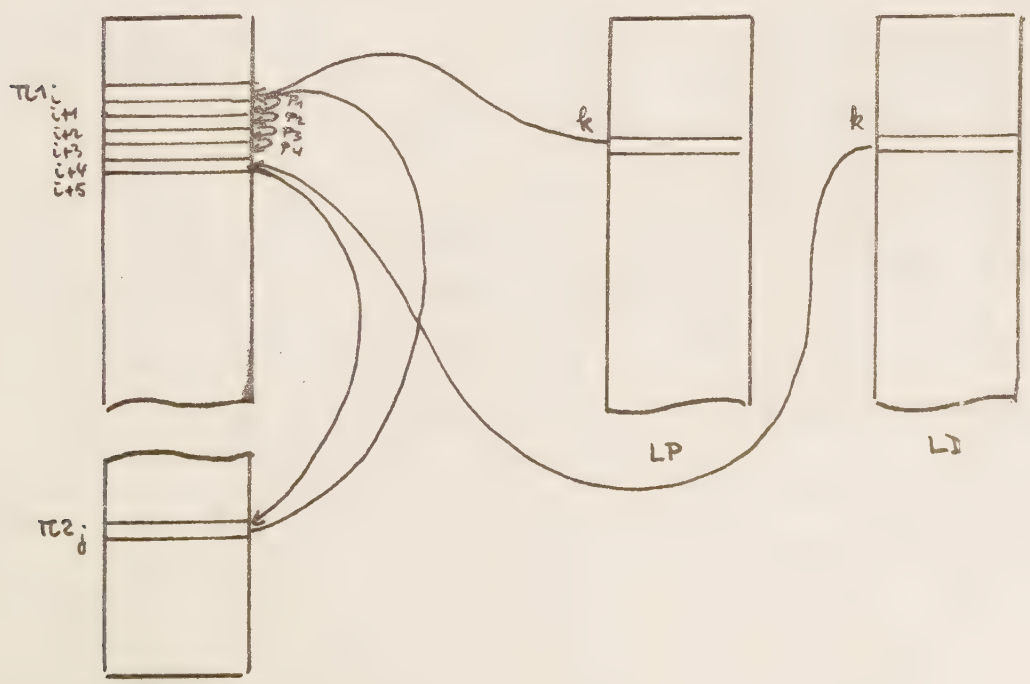
Pour cela nous avons utilisé une structure de piles avec multipointeurs. Les primitives π1 et π2 sont rangées séquentiellement en repérant les débuts et fins de lignes par les pointeurs LP et LD respectivement. Les π2 sont reliées aux π1 qui les composent par des pointeurs P_i.

Le croquis ci-dessous illustre ces dispositions en mémoire. On y voit par exemple les relations possibles entre les primitives π1 et π2.

L'élément j des π2 est constitué par 4 primitives de types π1.

Chaque élément $\pi 1_{i+1}$, $\pi 1_{i+2}$,..., $\pi 1_{i+4}$ est relié au suivant par un pointeur. Pour $\pi 1_{i+4}$ le pointeur P5 indique π2j, qui pointe lui-même sur $\pi 1_{i+1}$ grâce à P1.

Le pointeur P1 est initialisé dans le programme de concaténation qui crée les primitives π2. Les pointeurs P2, P3, P4 sont mis en place lors de l'appel du sous-programme ADDLOB. Quant à P5, c'est l'appel du sous-programme FINSUP qui provoque sa mise en place (voir description des programmes).



3.5. Descriptifs des programmes et résultats

- Programme principal. Signification et format des variables

VARIABLE	DIMENSIONS	FORMAT	SIGNIFICATION
ILOBE (K. L)	KMAXIM x LMAXIM	I2	Lobes de convolution
KLOBE (K. L)	KMAXIM x LMAXIM	I1	Lobes d'effacement
KOR (IDIR)	4	I2	Origines du lobe dans les
LOR (IDIR)	4	I2	tableaux ILOBE et KLOBE
KMAX (IDIR)	4	I2	Extrêmités au lobe
LMAX (IDIR)	4	I2	
IOR (IDIR)	4	I2	Coordonnées relatives du foyer
JOR (IDIR)	4	I2	dans le lobe
INOR (IDIR)	4	I2	Pas de progression du foyer
JNOR (IDIR)	4	I2	
I SEU (IS)	ISEUMX	I4	Tableau des différents seuils
MI (I, J)	IMAX x JMAX	I2	Image donnée
MA (i, j)			Image donnée (permettant de régénérer l'image MI après traitement)
ITOTMX			Nombre maximum de lobes
ITOT			Paramètre de rangement des lobes
ISEUDP			Niveau de départ
ISEC			Dimension de la marge
IDIREF			Direction du lobe trouvé
IDIRMX			Nombre maximum de directions (7)
KDIRMX			Numéro de la 1ère direction après la verticale (=5)
IDIRC			Direction symétrique d'une direction KDIRMX
ICENT (ITOT)	ITOTMX		Coordonnées du foyer du lobe ITOT
JCENT (ITOT)	ITOTMX		
K POIDS (ITOT)	ITOTMX		Poids du lobe ITOT
KDIR (ITOT)	ITOTMX		Direction du lobe ITOT
N			Numéro de crête
M			Numéro de lobe dans la crête
MN (N)	100		Nombre de lobes de la crête N
NBIMAG			Numéro de l'image traitée

- Sous-programme LOBMAX

- But :

Déterminer la possibilité de placer autour d'un point de coordonnées i, j un lobe dont le poids soit supérieur au seuil de filtrage. Effacement de la partie de l'image couverte par le lobe dans le cas d'acceptation.

- Variables utilisées :

Variable	Signification
En entrée	
i, j	Coordonnées du foyer du lobe dans l'image
En traitement	
IPOIDS	Poids du lobe courant
IDIR	Direction du lobe courant
IEFFAC	Paramètre d'effacement
En sortie	
ITROUV	= 1 si lobe trouvé = 0 sinon
IDIREF	Direction du lobe trouvé
IPOIRF	Poids du lobe trouvé.

- Sous-programme CALPOI

- But :

a) Si IEFFAC = 0

Calculer le poids d'un lobe de direction IDIR sont le foyer se trouve au point i, j.

b) Si IEFFAC = 1

Effacer les points utilisés par un lobe direction IDIR dont le foyer est en i, j

- Variables utilisées :

Variable	Signification
En entrée	
IEFFAC	Paramètre d'effacement
IDIR	Direction du lobe
I, J	Coordonnées du foyer
M1(I, J)	Image donnée
En traitement	
	Toutes les données relatives aux lobes
En sortie	
IPOIDS	Poids du lobe (si IEFFAC = 0)
M1(i, j)	Image modifiée (si IEFFAC = 1)

- Description du programme global

L'organigramme du programme se trouve en planche [II.6].

Le programme est décrit en FORTRAN 360/50.

Il lit un fichier disque FORTRAN O S qui comprend un enregistrement par lobe avec 8 quantités.

1. Numéro de la crête (I3)
2. Numéro du lobe dans la crête (I2)
3. Coordonnées de l'origine du lobe (2 I3)
4. Coordonnées de l'extrémité du lobe (2 I3)
5. Direction du lobe (I2)
6. Poids du lobe (I4)
7. Numéro d'image (I1)
8. Numéro du lobe dans l'image (I3)

Le programme lit une image en éliminant les crêtes de longueur inférieure à LMIN dans le sous-programme READ.

L'image est examinée crête par crête. Dès que 2 lobes consécutifs ont une variation de pente inférieure à DELDIR ou initialise un superlobe qui ne sera clos (dans FINSUP) que lorsqu'il aura atteint une rupture de pente supérieure à $\pm$ DELDIR par rapport au premier lobe du superlobe ou bien la fin de la crête. Chaque fois qu'on ajoute un lobe au superlobe on fait appel au sous-programme ADDLOB.

Les données du superlobe sont :

MPOIDS = somme des poids des lobes
MDIR = somme des directions des lobes
ICENT) = somme des coordonnées des lobes
JCENT)

Les lobes faisant partie d'une même superlobe pointent les uns sur les autres, le dernier pointant sur le superlobe et celui-ci pointant sur le premier pour faciliter les recherches.

Subroutine ADDLOB

augmente un superlobe existant KMLOB d'un lobe élémentaire.

Subroutine FINSUP

termine un superlobe (met le pointeur)

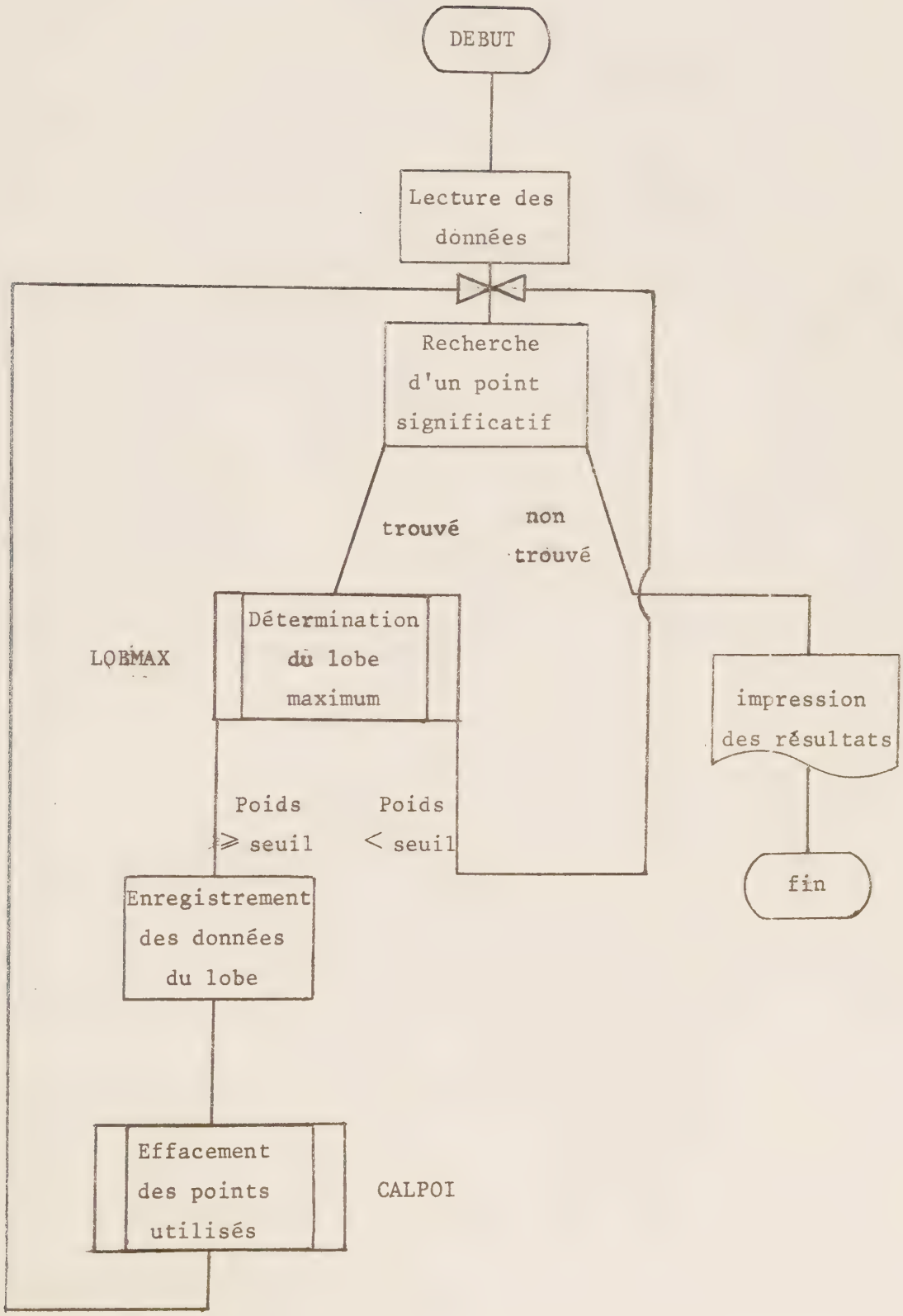
Subroutine READ

lit les lobes trouvés par le programme précédent sur un fichier disque en éliminant les lignes de longueur inférieure à LMIN.

Description des variables

KLOB - numéro du lobe courant traité
KSLOB - numéro du premier superlobe
KMLOB - nombre de lobes occupés parmi les KDLOB
KDLOB - nombre total de lobes disponibles (ici 500)
LOB (500, 6) - 500 lobes
LOB (X, 1) = MPOIDS (X) poids du lobe
LOB (X, 2) = MDIR (X) direction du lobe
LOB (X, 3) = IDENT (X)
LOB (X, 4) = JCENT (X) coordonnées du centre du lobe
LOB (X, 5) = MNB (X) longueur du lobe (0 pour un lobe élémentaire, = 0 pour un superlobe)
LOB (X, 6) = MLIEN (X) pointeur
LP (100) = premier lobe d'une crête
LD (100) = dernier lobe d'une crête
NCRET = nombre de crêtes
LONG = longueur d'un superlobe
IHIST (21) zone d'histogramme
MAX longueur de cette zone
LMIN largeur minimum d'une crête
IN numéro logique du fichier d'entrée
IMAG numéro d'image
DELDIR variation de direction admise pour lisser des lobes
KCRET crête courante

PLANCHE II.2.1



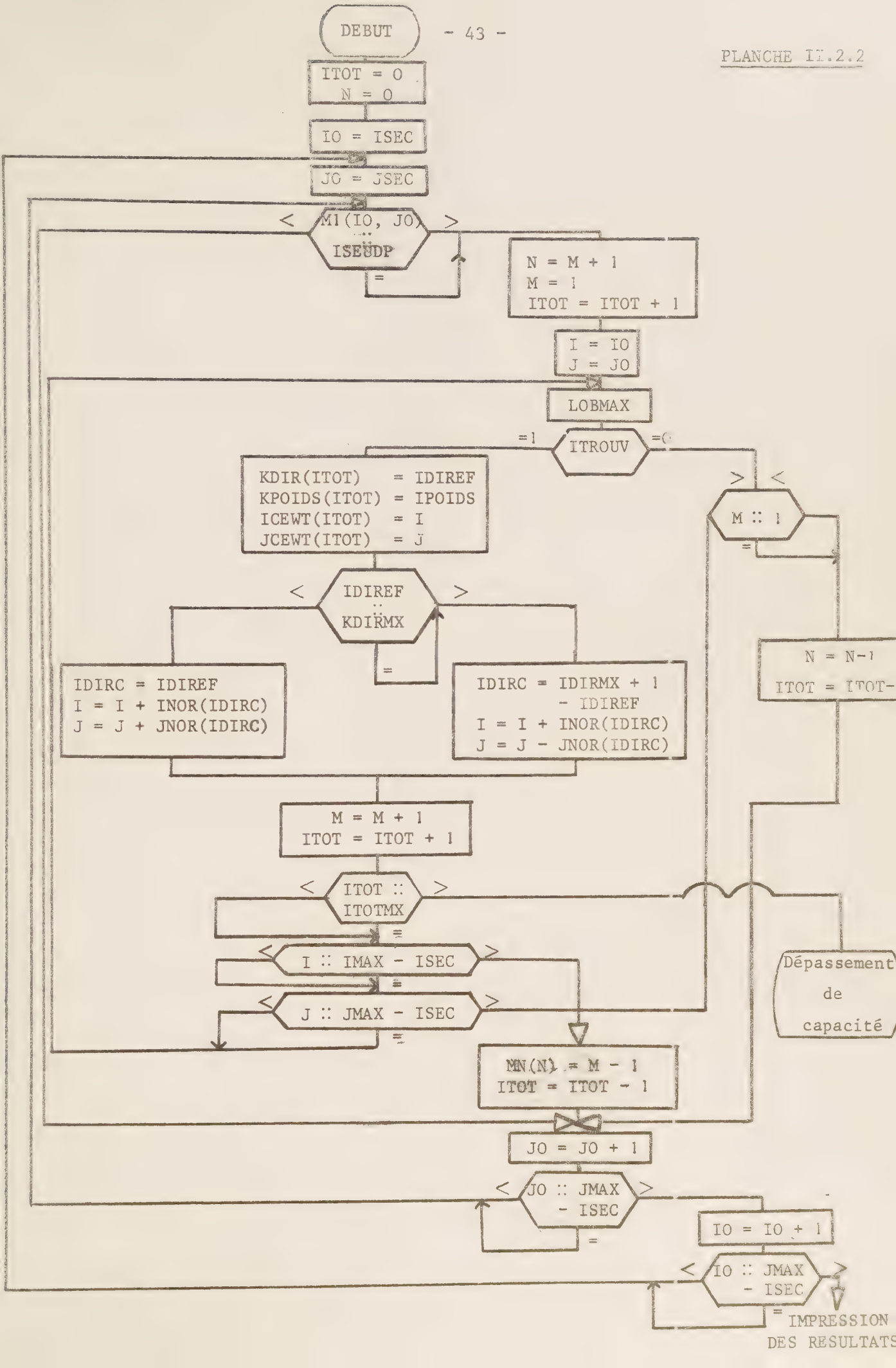

DEBUT
ITOT = 0
N = 0
IO = ISEC
JO = JSEC
M1(IO, JO) :: ISEUDP
N = M + 1
M = 1
ITOT = ITOT + 1
I = IO
J = JO
LOBMAX
ITROUV
KDIR(ITOT) = IDIREF
KPOIDS(ITOT) = IPOIDS
ICEWT(ITOT) = I
JCEWT(ITOT) = J
M :: 1
IDIREF :: KDIRMX
IDIRC = IDIREF
I = I + INOR(IDIRC)
J = J + JNOR(IDIRC)
IDIRC = IDIRMX + 1 - IDIREF
I = I + INOR(IDIRC)
J = J - JNOR(IDIRC)
N = N-1
ITOT = ITOT-
M = M + 1
ITOT = ITOT + 1
ITOT :: ITOTMX
I :: IMAX - ISEC
J :: JMAX - ISEC
Dépassement de capacité
MN(N) = M - 1
ITOT = ITOT - 1
JO = JO + 1
JO :: JMAX - ISEC
IO = IO + 1
IO :: JMAX - ISEC
IMPRESSION DES RESULTATS

PLANCHE II.2.3

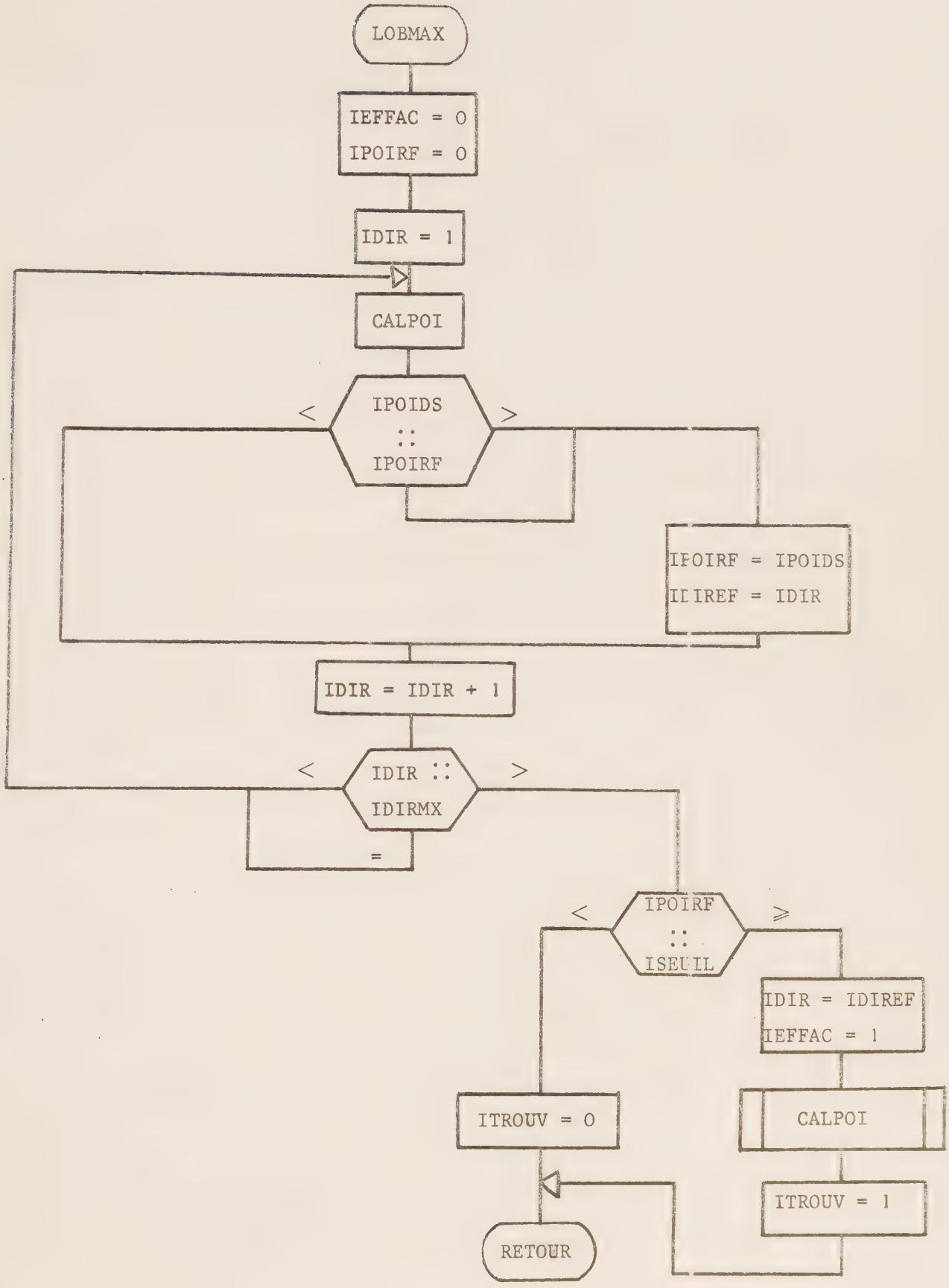
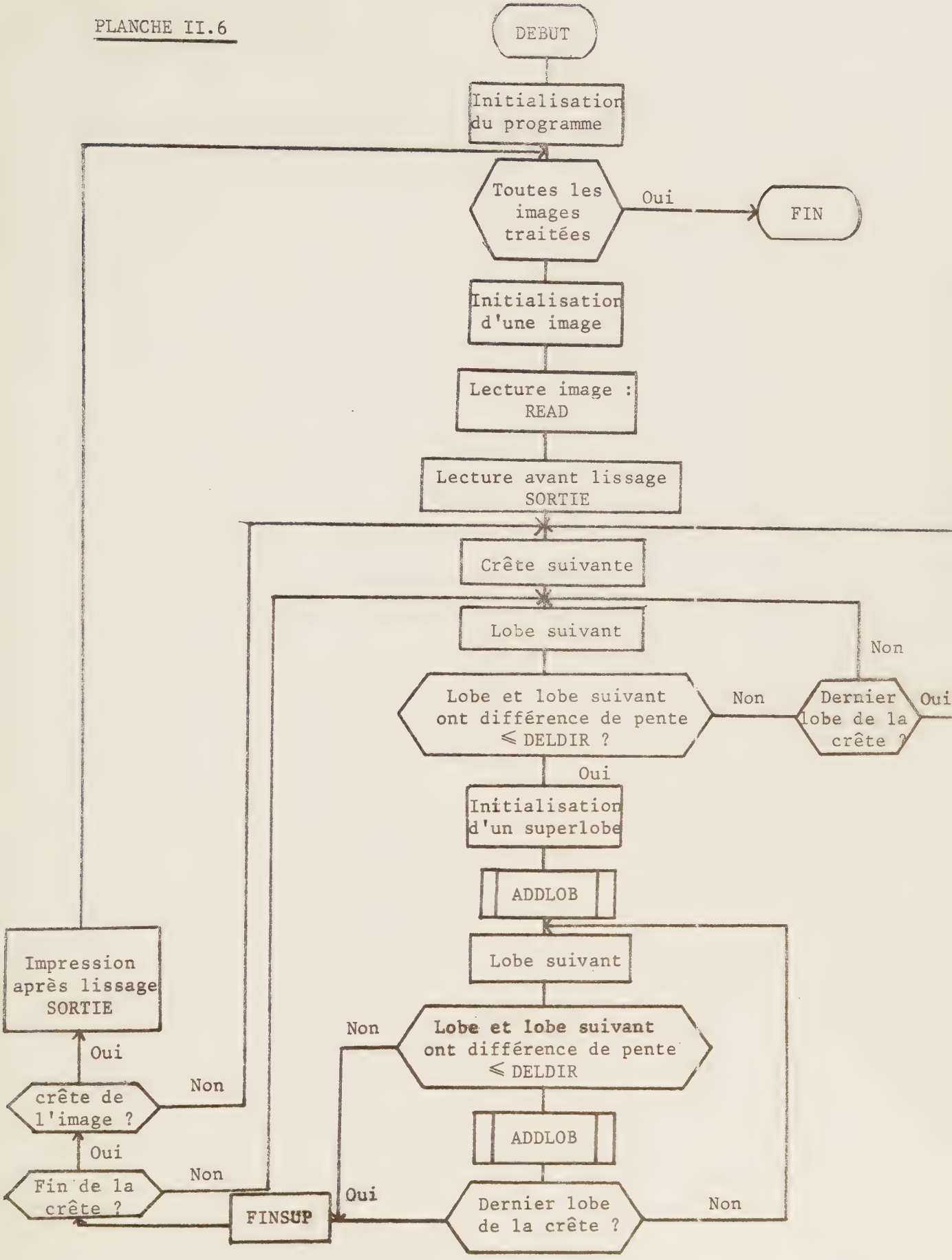


PLANCHE II.6


DEBUT
Initialisation du programme
Toutes les images traitées
Oui
FIN
Initialisation d'une image
Lecture image : READ
Lecture avant lissage SORTIE
Crête suivante
Lobe suivant
Non
Lobe et lobe suivant ont différence de pente ⩽ DELDIR ?
Non
Dernier lobe de la crête ?
Oui
Oui
Initialisation d'un superlobe
ADDLOB
Lobe suivant
Non
Lobe et lobe suivant ont différence de pente ⩽ DELDIR
ADDLOB
Dernier lobe de la crête ?
Oui
Non
FINSUP
Fin de la crête ?
Non
Oui
crête de l'image ?
Non
Oui
Impression après lissage SORTIE
Oui

"LISTINGS" DES PROGRAMMES

```
//STEF1108 JOB HN1108,STEFFEN,MSGLEVEL=1,CLASS=D
//STEP1 EXEC FTLIBCLG,PARM.FORT=BCD
//FORT.SYSIN DD *
      COMMON KLOB,KSLOB,KMLOB,KDLOB,LOB(500,6),LP(100),LD(100),NCRET
      COMMON LONG,MAX,IHIST%21<,LMIN,IN,IMAG
      COMMON BUFFER%200<
      DIMENSION  MPOIDS(500),MDIR(500),ICENT(500),
     1   JCENT(500),MLIEN(500),MNB(500)
      EQUIVALENCE (LOB(1,1),MPOIDS),(LOB(1,2),MDIR),(LOB(1,3),ICENT),
     1   (LOB(1,4),JCENT),(LOB(1,5),MNB),(LOB(1,6),MLIEN)
C     DATA DELDIR,KDLOB,MAX,LMIN,IN/1,500,21,3,16/
C CES CARTES REMPLACENT LA PRECEDENTE POUR ETRE COMPATIBLE I.B.M.
      DELDIR = 1
      KDLOB = 500
      MAX = 21
      LMIN = 3
      IN = 16

C  IN EST LE NUMERO LOGIQUE D@ENTREE
      IMAG # 1
       CALL PLOTS%BUFFER,800<
1     IF% IMAG .GT. 2 < GO TO  1200
C  POUR TRAITER DEUX IMAGES
      KSLOB # 0
      NCRET # 1
      LP%1< # 1

      DO 10 K#1,KDLOB
      MNB%K< # 0
      MLIEN%K< # 0
10    CONTINUE

      CALL READ

      KMLOB=KSLOB

      DO 20 K#1,MAX
      IHIST%K< # 0
20    CONTINUE

      CALL SORTIE

      DO 1000 KCRET=1,NCRET
      KP=LP(KCRET)
      KD=LD(KCRET)
      KDM1=KD-1
50    DO 100 KLOB=KP,KDM1
      IF(IABS( MDIR(KLOB) - MDIR(KLOB+1) ) .LE. DELDIR ) GO TO 200
100   CONTINUE
      GO TO 1000
200   KMLOB =KMLOB + 1
      IF( KMLOB .GT. KDLOB ) GO TO 1100
      DO 220 K=1,4
      LOB(KMLOB,K) = LOB(KLOB,K)
220   CONTINUE
      MLIEN(KMLOB) = KLOB
      MNB(KMLOB) = 1
      KFIRST = KLOB
```

```
      KLOB = KLOB + 1
      LONG # 1
      CALL ADDLOB
      KP = KLOB + 1
      IF(KP.GT.KD) GO TO 350
      DO 300 KLOB = KP,KD
      IF(IABS( MDIR(KFIRST) - MDIR(KLOB) ) .GT. DELDIR ) GO TO 400
      CALL ADDLOB
300   CONTINUE
C POUR QUE FINSUP MARCHE SI LA BOUCLE SE TERMINE
350   KLOB # KLOB & 1
400   CALL FINSUP
      IF( KLOB .GE. KD ) GO TO 1000
      KP = KLOB
      GO TO 50
1000  CONTINUE
1100  CALL SORTIE
      CALL GRAPH
      IMAG # IMAG & 1
      GO TO 1
1200  CALL PLOT%0.,0.,444<
      CALL EXIT
      END
      SUBROUTINE ADDLOB
      COMMON KLOB,KSLOB,KMLOB,KDLOB,LOB(500,6),LP(100),LD(100),NCRET
      COMMON LONG,MAX,IHIST%21<,LMIN,IN,IMAG
      DIMENSION  MPOIDS(500),MDIR(500),ICENT(500),
     1   JCENT(500),MLIEN(500),MNB(500)
      EQUIVALENCE (LOB(1,1),MPOIDS),(LOB(1,2),MDIR),(LOB(1,3),ICENT),
     1   (LOB(1,4),JCENT),(LOB(1,5),MNB),(LOB(1,6),MLIEN)
      DO 10 K=1,4
      LOB(KMLOB,K) = LOB(KMLOB,K) + LOB(KLOB,K)
10    CONTINUE
      MLIEN(KLOB-1) = KLOB
      MNB(KMLOB) = MNB(KMLOB) + 1
      LONG # LONG & 1
      RETURN
      END
      SUBROUTINE FINSUP
      COMMON KLOB,KSLOB,KMLOB,KDLOB,LOB(500,6),LP(100),LD(100),NCRET
      COMMON LONG,MAX,IHIST%21<,LMIN,IN,IMAG
      DIMENSION  MPOIDS(500),MDIR(500),ICENT(500),
     1   JCENT(500),MLIEN(500),MNB(500)
      EQUIVALENCE (LOB(1,1),MPOIDS),(LOB(1,2),MDIR),(LOB(1,3),ICENT),
     1   (LOB(1,4),JCENT),(LOB(1,5),MNB),(LOB(1,6),MLIEN)
      MLIEN(KLOB-1) = KMLOB
      IF%LONG .LT. MAX < GO TO 100
      IHIST%MAX< # IHIST%MAX< & 1
      GO TO 200
100   IHIST%LONG< # IHIST%LONG< & 1
200   CONTINUE
      RETURN
      END
      SUBROUTINE SORTIE
      COMMON KLOB,KSLOB,KMLOB,KDLOB,LOB(500,6),LP(100),LD(100),NCRET
      COMMON LONG,MAX,IHIST%21<,LMIN,IN,IMAG
      DIMENSION  MPOIDS(500),MDIR(500),ICENT(500),
     1   JCENT(500),MLIEN(500),MNB(500)
```

```
      EQUIVALENCE (LOB(1,1),MPOIDS),(LOB(1,2),MDIR),(LOB(1,3),ICENT),
     1   (LOB(1,4),JCENT),(LOB(1,5),MNB),(LOB(1,6),MLIEN)
      PRINT 1000,NCRET,(I,LP(I),LD(I),I=1,NCRET)
      PRINT 1100,KSLOB,KMLOB,KDLOB,(J,(LOB(J,I),I=1,6),J=1,KMLOB)
      PRINT 1200,IHIST
      RETURN
1000  FORMAT(1H1,I10//(1H ,3I10))
1100  FORMAT(1H0,3I10//(1H ,7I10))
1200  FORMAT%1H0,21I4<
      END
      SUBROUTINE READ
      COMMON KLOB,KSLOB,KMLOB,KDLOB,LOB(500,6),LP(100),LD(100),NCRET
      COMMON LONG,MAX,IHIST%21<,LMIN,IN,IMAG
      DIMENSION  MPOIDS(500),MDIR(500),ICENT(500),
     1   JCENT(500),MLIEN(500),MNB(500)
      EQUIVALENCE (LOB(1,1),MPOIDS),(LOB(1,2),MDIR),(LOB(1,3),ICENT),
     1   (LOB(1,4),JCENT),(LOB(1,5),MNB),(LOB(1,6),MLIEN)
      NLOB # 0
      NCR # 1
      GO TO 100

90    IF% N.NE. IMAG < GO TO 350
      NLOB # 1

100   KSLOB # KSLOB & 1
      IF% KSLOB .GT. KDLOB < GO TO 400
      NLOB # NLOB & 1
      READ(IN,1000,END=350)
     1               NCLU,ICENT(KSLOB),JCENT(KSLOB),MDIR(KSLOB),
     2       MPOIDS(KSLOB),N

C     READ%IN,1000< NCLU,ICENT%KSLOB<,JCENT%KSLOB<,MDIR%KSLOB<,
C    1       MPOIDS%KSLOB<,N

      IF% NCR .EQ. NCLU < GO TO 100

      NCR # NCLU
      IF% NLOB .GT. LMIN < GO TO 300

      KSLOB # KSLOB - %NLOB-1<
      DO 210 K#1,4
      LOB%KSLOB,K< # LOB% KSLOB&NLOB-1, K <
210   CONTINUE
      GO TO 90

300   LD%NCRET< # KSLOB - 1
      NCRET # NCRET & 1
      IF% NCRET .GT. 100 < GO TO 420
      LP% NCRET < # KSLOB
      GO TO 90

350   KSLOB # KSLOB - 1
      NCRET # NCRET - 1
      RETURN

400   PRINT 1100,KDLOB
      CALL EXIT
```

```
420   PRINT 1200
      CALL EXIT
1000  FORMAT(I3,2X,2I3,6X,I2,I4,     I1)
1100  FORMAT('0PLUS DE'I4,' LOBES')
1200  FORMAT('0PLUS DE 100 CRETES')
      END
      SUBROUTINE GRAPH
      COMMON KLOB,KSLOB,KMLOB,KDLOB,LOB(500,6),LP(100),LD(100),NCRET
      COMMON LONG,MAX,IHIST%21<,LMIN,IN,IMAG
      COMMON BUFFER%200<
      DIMENSION  MPOIDS(500),MDIR(500),ICENT(500),
     1   JCENT(500),MLIEN(500),MNB(500)
      EQUIVALENCE (LOB(1,1),MPOIDS),(LOB(1,2),MDIR),(LOB(1,3),ICENT),
     1   (LOB(1,4),JCENT),(LOB(1,5),MNB),(LOB(1,6),MLIEN)
      DIMENSION X%4<,Y%4<
      PI4 # 3.14159/4.
      CALL PLOT%2.,2.,3<
      CALL PLOT%35.,2.,2<
      CALL PLOT%35.,24.5,2<
      CALL PLOT%2.,24.5,2<
      CALL PLOT%2.,2.,2<
      CALL PLOT%2.5,2.5,-3<
      X(3) = 0.
      Y(3) = 0.
      X%4< #1.
      Y%4< # 1.
      DO 500 KCRET # 1,NCRET
      KP # LP%KCRET<
      KD # LD%KCRET<
      X%2< # ICENT%KP</4.
      Y%2< # JCENT%KP</4.
      CALL NUMBER%X%2<,Y%2<,0.1,8.,0.,-1<
      KP # KP & 1
      DO 400 KLOB # KP,KD
      X%1< # X%2<
      Y%1< # Y%2<
      X%2< # ICENT%KLOB</4.
      Y%2< # JCENT%KLOB</4.
      CALL LINE(X,Y,2,1,0,13)
      IF% MLIEN%KLOB-1< .NE. KLOB < CALL NUMBER%X%2<,Y%2<,0.1,8.,0.,-1<
400   CONTINUE
      X%1< # X%2<
      Y%1< # Y%2<
      TETA # %4 - MDIR%KD< < * PI4 / 3.
      X%2< # X%2< & 1. * COS% TETA <
      Y%2< # Y%2< & 1. * SIN% TETA <
      CALL LINE(X,Y,2,1,0,13)
      CALL NUMBER% X%2<,Y%2<,0.1,8.,0.,-1<
500   CONTINUE
      CALL PLOT%60.,0.,-3<
      RETURN
      END
/*
//GO.FT16F001 DD DSN=LACROIX.X1108.PGLOBE,DISP=OLD,UNIT=SYSDA,          X
//       VOL=SER=PETIT1
//GO.PLOTTERC DD UNIT=2400-3,LABEL=(1,NL),DSN=PTDCB,DCB=(DEN=3,RECFM=U,*
//       BLKSIZE=800),VOLUME=SER=TRACE2
//GO.SYSABEND DD SYSOUT=A
```

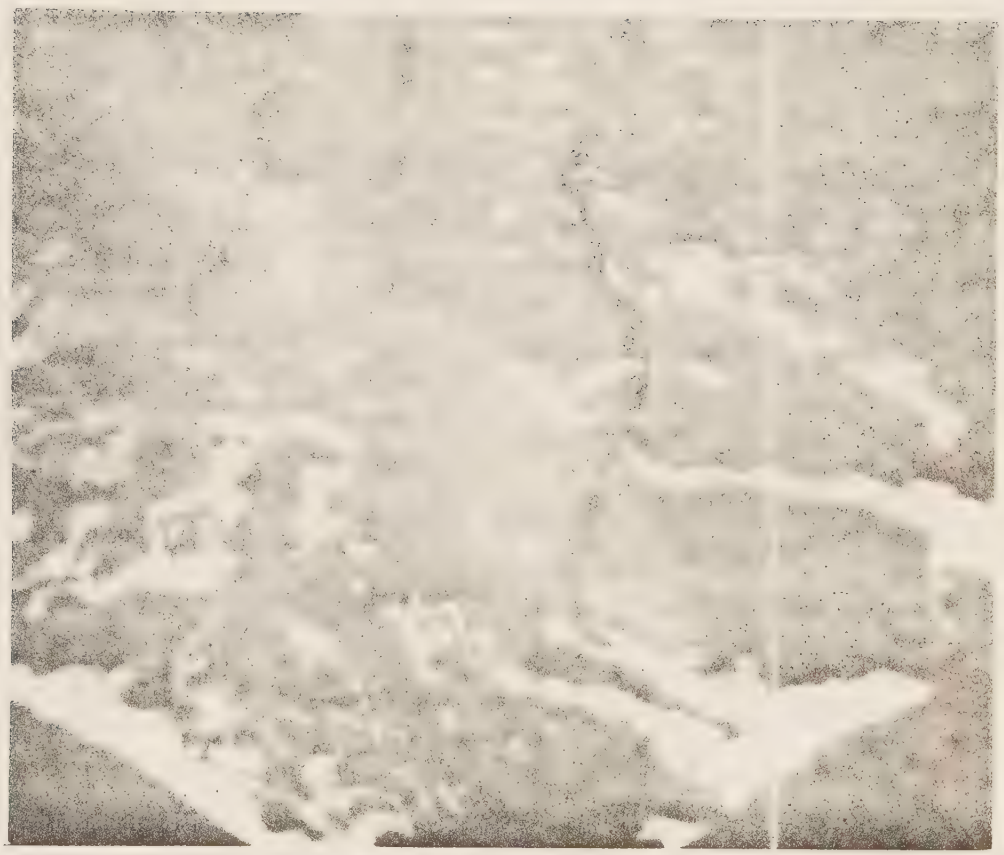


Planche II o 1

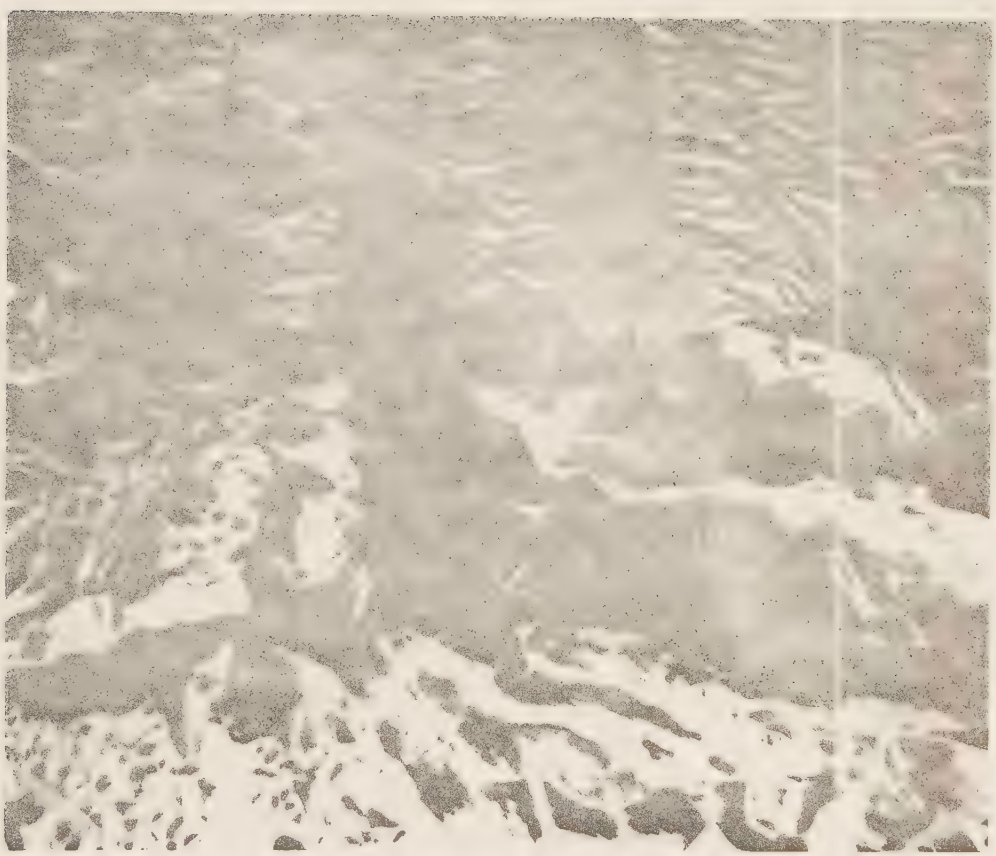


Planche II o 2

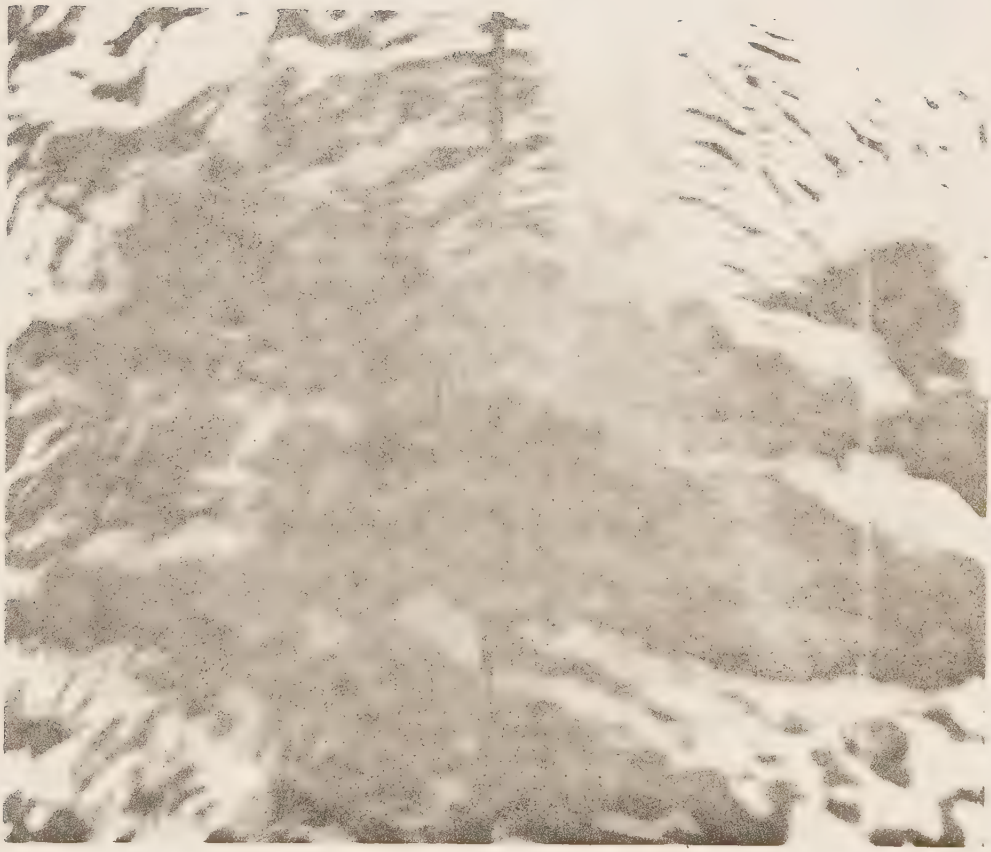


Planche II 03

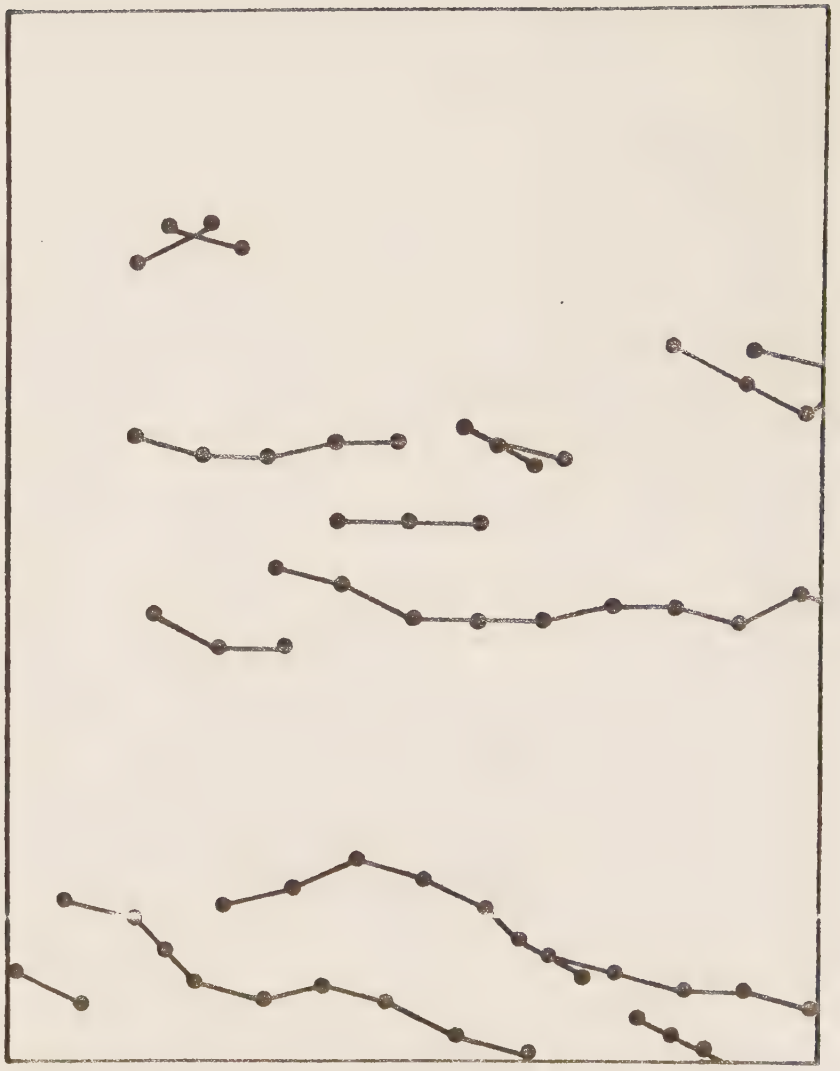
DONNEE RADAR

DONNEE PREVISIONNELLE

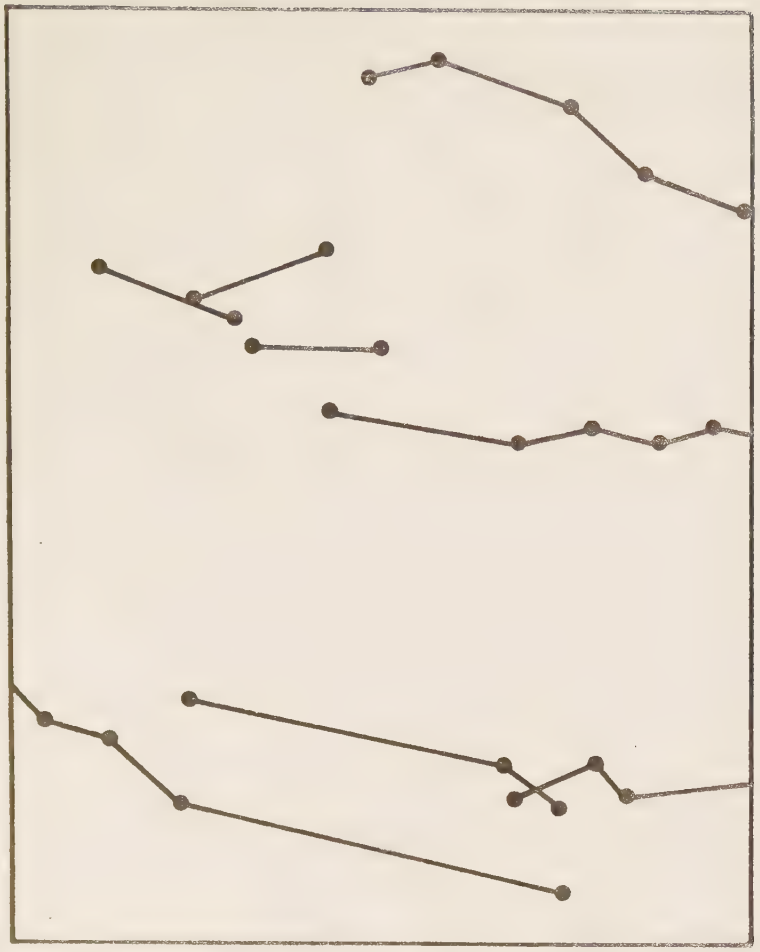
DONNEE PREVIS. DISCRETISEE

DONNEE RADAR DISCRETISEE

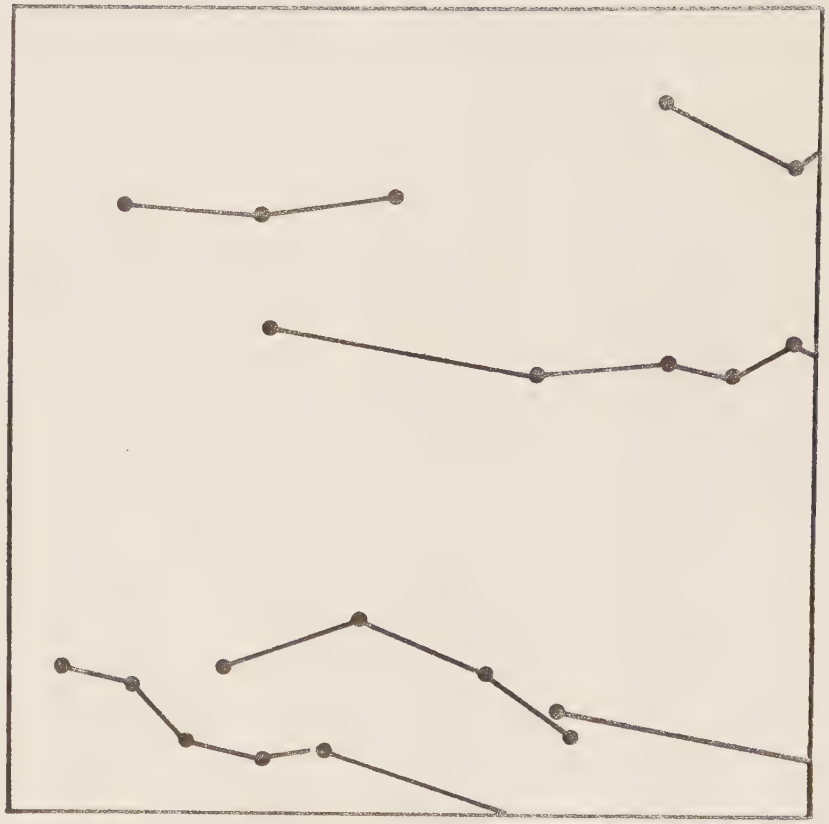
DONNEE RADAR TRAITEE



DONNEE PREVISIONNELLE TRAITEE



SCOPE RADAR



PREVISIONNELLE

LISSAGE

PLANCHE 4

Bibliographie relative aux chapitres I et II

[1] J. ARSAC et J. C. SIMON : Représentation d'un phénomène physique par une somme de translatées, Annales de Radioélectricité, 15 n° 61, juillet 60.

[2] J. P. AUBINEAU et J. C. SIMON : Function and signal representation in connection with precision. Institut de Programmation - Paris - Août 1970.

[3] R. HAOUDAS et J. C. HORVAT : Representation d'une fonction par une somme de translatées. Institut de Programmation - Paris - Juin 1970.

[4] G. GUIHO et J. C. SIMON : Picture representation and transformations by computer in pattern Recognition - Décembre 1971 (PRS).

[5] J. ARSAC : La Transformée de Fourier et ses applications (Dunod edit.)

[6] F. F. KUO et J. F. KAISER : System Analysis by Digital Computers (Wiley edit.)

[7] J. A. DUNNE : Mariner 1969 Television Image Processing. Pattern Recognition, 2 pp. 261-268, Décembre 1970.

[8] Ed. KAHN : Thèse de Doctorat 3ème cycle - Faculté des Sciences de Paris Décembre 1970.

[9] J. C. SIMON et A. CHECROUN : "Reconnaissance de formes à deux dimensions C. R. Acad. Sc. Paris, t. 268, pp. 967-970 - 28 avril 1969.

[10] R. NATHAN : "Picture Enhancement for the Moon, Mars and Man" in Pictorial Pattern Recognition. Thomson Book Company - 1968.

[11] J. P. FAIL et G. GRAU : Les filtres en éventail. Geophysical prospecting, 11, # 2, pp. 131-163, 1970.

[12] P. BOIS et M. LA PORTE : Pointé automatique. Geophysical prospecting 18, # 4, pp. 489-504, 1970.

[13] G. GUIHO : Thèse de 3ème cycle - Université Paris VI.

[14] A. CHECROUN et C. ROCHE : "La notion de Squelette en Reconnaissance des Formes" Revue Automatisme.

[15] J. Mc CARTHY et al : A computer with hands eyes and ears. AFIPS Conferences Proc., vol 33 (1968).

[16] S. TSUJI et al : Industrial Eye. Conf. of Inst. of Electric. Eng. (1970).

[17] A. ROSENFELD : Picture Processing by Computer. Academic Press. New-York (1969).

[18] A. GUZM'AN : Computer Recognition of three-dimensional Objects in visual scenes. Ph. D. Thesis. M. I. T. (Dec. 1968).

[19] U. MONTANARI : On the optimal detection of curves in noisy pictures. C. A. C. M. 14, N. 5, (may 71).

[20] J. T. TOU : On Feature Encoding in Picture Processing by Computer. Proc. of the Seventh Annal Allerton Conf. University of Illinois (1969).

CHAPITRE III

Opérateurs de classement des primitives

§1. UTILISATION DE LA NOTION DE DISTANCE POUR REGROUPER DES ELEMENTS COMPARABLES

1.1. Définitions et rappels

A un niveau déterminé les formes pertinentes ou caractéristiques qui ont été reconnues par des traitements antérieurs deviennent "primitives" pour les opérations suivantes. Elles sont affectées d'un nom et éventuellement de certaines valeurs numériques.

A la sortie du capteur on peut observer le premier niveau de formes caractéristiques : ce sont les résultats fournis par une rétine par exemple. La reconnaissance des lignes fournit un deuxième niveau de primitives. Les niveaux suivants peuvent consister par exemple à distinguer plusieurs formes de lignes, puis des groupements de lignes etc... L'illustration la meilleure est offerte par la reconnaissance des caractères : on pourra parler alors du niveau des signes, du niveau des mots ou du niveau des groupes de mots...

Cette hiérarchisation des processus de reconnaissance semble nécessaire dès qu'on absorbe des problèmes complexes.

H. SIMON dans "The Sciences of Artificial" (MIT Press 1970) ≪ affirme que ≪ ... le fait que la plupart des systèmes complexes ont une ≫ ≪ structure hiérarchique est un facteur essentiel pour comprendre, décrire ≫ ≪ et même "voir" l'ensemble et les parties ≫ .

Quel que soit le système de reconnaissance que l'on considère, il est donc peut être intéressant de dégager a priori l'existence de certains niveaux hiérarchiques. Nous allons considérer un certain nombre de méthodes classiques de construction de hiérarchies, puis nous donnerons une application de ces méthodes aux primitives du problème du recalage radar.

1.2. La notion de distance entre éléments comparables

On peut dire d'abord que le fait de décider qu'un élément appartient ou non à une classe est le résultat d'une opération concrète de reconnaissanc La répartition d'éléments dans des classes distinctes ne peut évidemment se faire que si ces éléments sont comparables et séparables.

Considérons alors un ensemble X d'éléments comparables dont les noms i, j, k... sont affectés de certains paramètres. Supposons que grâce à ces paramètres on puisse définir une distance d(i, j) entre chaque couple {i, j}. Autrement dit, on suppose qu'on a pu trouver une fonction $d \in \mathbb{R}$ telle que :

$$d \geqslant 0$$

$$d(i, j) = d(j, i)$$

$$d(i, i) = 0$$

Et enfin, $$d(i, j) \leqslant d(i, k) + d(j, k)$$

Si cette distance est normalisée (ramenée à l'intervalle [0, 1]) on dit que c'est une mesure de dissemblance.

La fonction complémentaire s = 1 - d est appelée mesure de ressemblance ou indice de similarité.

1.3. Les procédés d'agglutination ou Clustering

Soit E un ensemble fini d'éléments {i} et H un ensemble de parties de E($H \in \mathcal{P}(E)$).

Si l'ensemble H est défini a priori, les éléments de E pourront être rattachés à ceux de H de telle manière que les éléments d'une même classe se "ressemblent" plus que ceux appartenant à deux classes différentes. C'est ce qu'on a appelé "Clustering" ou "Procédé d'agglutination". L'algorithme général est le suivant :

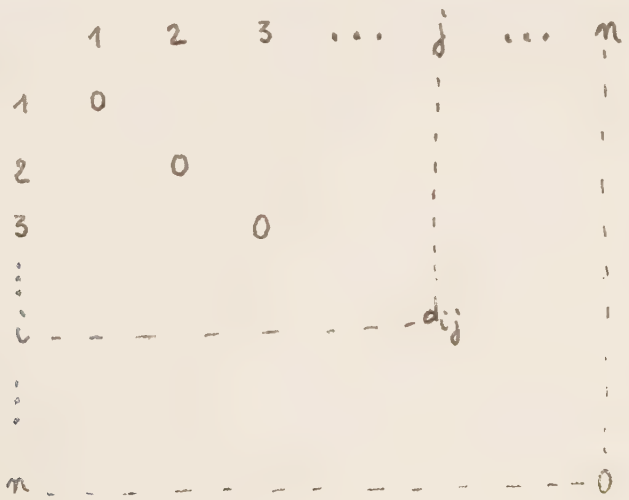
1. Définir N classes a priori (par exemple au hasard).
2. Si D(i, K) est la distance de l'élément i à la classe K ($i \in E$, $K \in H$), on choisit la plus petite parmi les N distances de i aux classes de H. Alors si $D(i, L) = \inf_{K \in [1,N]} D(i, K)$, on rattache i à la classe L. On recommence pour tous les i.
3. La partition est-elle changée. Si oui retour à 2 sinon arrêt.

Autour de cet algorithme, parmi les quelques variations que nous connaissons, nous citerons les travaux de DIDAY [2]. Ce dernier propose de définir chaque classe $K \in H$ par n_K éléments "caractéristiques" (Etalons). Ces étalons forment un sous-ensemble E_K et la distance D utilisés dans l'algorithme est la distance de i à E_K. Le choix des n_K éléments se fait à la fin de phase (1) en minimisant une fonction R(i, K, L) où L est l'ensemble des E_K tel qu'il se trouve à la fin de la phase (1).

Sous des conditions très générales sur R, l'algorithme semble bien converger avec un temps de calcul et une occupation de mémoire raisonnable. Nous essayons actuellement d'appliquer cet algorithme pour guider à la définition de primitives dans le problème des cartes radar.

1.4. Les ordonnances

Si l'ensemble E possède n éléments i, j, k, ... il existe $\frac{n(n-1)}{2}$ couples {i, j} dont les distances si elles sont données, peuvent se représenter dans un tableau triangulaire à double entrée. La diagonale étant formée de 0, à l'intersection de la ligne i et de la colonne j figure la distance ou mesure de dissemblance d_{ij} de l'élément i avec l'élément j.



Si d_{ij} vérifie les axiomes de distance, il est toujours possible d'établir une relation d'ordre total sur l'ensemble $\{d_{ij}\}$. Une bijection entre les d_{ij} et les couples ij permet alors de définir également une relation d'ordre total sur l'ensemble des couples $\{i, j\}$.

On dit alors que l'ensemble E est muni d'une "ordonnance". Alors quels que soient $j, k, l, m \in E$, l'une ou l'autre des inégalités suivantes est vérifiée :

$$\{j, k\} < \{l, m\} \quad \text{ou} \quad \{l, m\} < \{j, k\}$$

Dans le cas où l'on a un préordre sur E et non pas un ordre total, alors une troisième possibilité intervient :

$$\{l, m\} = \{j, k\}$$

Ce dernier cas (égalité) implique une notion de précision sur la détermination des mesures de dissemblances. L'égalité n'est alors plus une opération transitive, c'est-à-dire que :

$$\{i, j\} = \{k, l\} \quad \text{et} \quad \{k, l\} = \{m, n\}$$

n'entraîne pas que $\{i, j\} = \{m, n\}$

Nous reviendrons avec plus de détails sur la notion d'"ordonnance" et ses extensions au chapitre IV où nous utilisons des "ordonnances parenthèsées" pour comparer entre elles des figures de points.

1.5. Les hiérarchies[1]

Considérons à nouveau l'ensemble E = {i} et l'ensemble $H \in \mathcal{P}(E)$.

On dira que H est une hiérarchie sur E si :

- $E \in H$
- $\forall i \in E$ on a $\{i\} \in H$
- $\forall h$ et $h' \in H$ si $h \cap h' \neq \emptyset$ alors on ne peut avoir que $h \subset h'$ ou $h' \subset h$

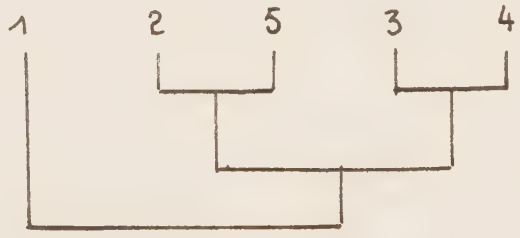
Le couple (E, H) formé d'un ensemble fini d'éléments et d'une hiérarchie sur cet ensemble peut se représenter par un graphe d'arbre (ou arborescence) dont les noeuds sont les parties h, h'... de E appartenant à H et dont les sommets pendants sont les éléments de E.

Par exemple, si E = {1, 2, 3, 4, 5} et si $H = \{h_1, h_2, \ldots, h_9\}$

avec $h_1 = \{1\}$ $h_2 = \{2\}$ $h_3 = \{3\}$ $h_4 = \{4\}$ $h_5 = \{5\}$

$h_6 = \{2, 5\}$ $h_7 = \{3, 4\}$ $h_8 = \{2, 3, 4, 5\}$ et $h_9 = E$

l'arbre représentatif de cette hiérarchie sera :



(1) Nous renvoyons pour les notions qui vont suivre à la thèse de M. ROUX [1]

On peut alors remarquer que l'on peut comparer entre elles certaines parties de E par exemple $h_6 < h_8$, mais que d'autres comme h_6 et h_7 ne peuvent pas être comparées.

La hiérarchie induit une relation d'ordre, en général non total, sur les paires d'éléments de E.

Hiérarchie indicée

Une hiérarchie est dite "indicée" si on a pu définir une application $x : H \to [0, 1]$ telle que :

- si $h \in H$ n'a qu'un élément $x(h) = 1$
- si $h \subset h' \in H$ alors $x(h) > x(h')$

C'est-à-dire que chaque niveau de l'arbre peut être affecté d'un indice $\in [0, 1]$; la valeur 0 étant celle de la racine (ensemble E complet) et la valeur 1 étant affectée au sommet (ensemble des parties à un élément).

La notion "d'indice de similarité"

Soient $k, l \in E$ et $\sigma(k, l)$ le plus grand nombre tel qu'il existe $h \in H$ avec $k, l \in h$ et $\sigma(k, l) = x(h)$. Cela revient à dire que σ correspond au niveau le plus élevé de l'arbre où se trouvent réunis encore les éléments k et l.

Alors on peut écrire que $\sigma(i, j) < \sigma(k, l) \Rightarrow \{i, j\} < \{k, l\}$

et $\sigma(i, j) = \sigma(r, s) \Rightarrow \{i, j\} = \{r, s\}$

σ définit donc une "mesure de ressemblance" ou encore un "indice de similarité" d'un couple d'éléments de E.

La quantité $\delta = 1 - \sigma$ possède des propriétés particulières :

- c'est une ultra distance, c'est-à-dire qu'elles vérifie les axiomes suivants :

$$\delta(i, j) \geqslant 0$$

$$\delta(i, j) = \delta(j, i)$$

et surtout $\quad \delta(i, j) \leqslant \text{Sup}\{\delta(i, k), \delta(k, j)\}$

On définit ainsi une "ultramétrique" sur E.

- Inversement, à toute ultramétrique sur E, on peut faire correspondre une hiérarchie indicée unique H.

Les notions de hiérarchie indicée et d'ultramétrique sont équivalentes.

Remarque 1 : dans un espace muni d'une ultradistance les triangles sont tous équilatéraux ou isocèles à base plus petite que les côtés.

Une métrique sur un ensemble d'objets comparables permet donc le regroupement automatique en classes disjointes. Ce qui est un but partiel de la Reconnaissance des Formes (il faut encore comparer ces classes aux éléments mémoires). Il est possible de présenter ces regroupements comme la création objective de "concepts", noms donnés aux classes trouvées. Sans informations a priori les éléments "concepts" ainsi trouvés sont conservés en mémoire et serviront aux futures opérations de reconnaissance.

Remarque 2 : l'inégalité vérifiée par une ultradistance est plus restrictive que celle vérifiée par une distance. En effet :

$$\delta(i, j) \leqslant \text{Sup}\,[\,\delta(i, k), \delta(k, j)] \;\Rightarrow\; \delta_{(i, j)} \leqslant \delta_{(i, k)} + \delta_{(k, j)}$$

alors que l'inverse n'est pas vrai.

Nous verrons qu'il est possible de construire une hiérarchie en passant d'une métrique à une ultramétrique particulière.

Ultramétrique sous dominante d'une métrique donnée

Deux métriques sont dites "équivalentes" si elles induisent sur un ensemble E la même ordonnance. L'ensemble des métriques équivalentes est infini.

De la même façon, deux ultramétriques sont équivalentes si elles induisent la même hiérarchie sur un ensemble E.

Ainsi, la donnée d'une métrique ou d'une ultramétrique induit une ordonnance ou une hiérarchie unique, mais inversement, on peut obtenir la même ordonnance ou la même hiérarchie à partir d'une infinité de métriques ou d'ultramétriques.

On peut donc envisager l'utilisation de ces notions en Reconnaissance des Formes, afin de manipuler des ensembles d'objets similaires et non plus des individus. C'est-à-dire que nous voyons là un moyen pratique d'introduire la notion de concept de manière plus ou moins objective.

En général, le problème est de passer d'une métrique quelconque à une hiérarchie sur l'ensemble donné. Pour que la hiérarchie obtenue soit unique, ROUX propose de construire une ultramétrique particulière, qui est la borne supérieure de l'ensemble des métriques équivalentes (c'est-à-dire qui induisent la même ordonnance sur E). Il l'appelle l'ultramétrique sous-dominante, démontre qu'elle est unique, et propose l'algorithme qui permet de la construire à partir du tableau des $\frac{n(n-1)}{2}$ distances données.

§2. UNE METHODE DE REGROUPEMENT

Nous inspirant des méthodes proposées par ROUX, nous avons mis au point la méthode ci-dessous de regroupement :

On considère un ensemble $E = \{e_i\}$ fini d'éléments comparables.

On suppose connue une métrique d et par suite on peut définir les distances d_{ij} des couples $\{e_i, e_j\}$ ou encore les indices de similarité $s_{ij} = 1 - d_{ij}$.

On peut donc supposer donnée la matrice $\{s_{ij}\}$ des indices de similarité entre les couples d'éléments (e_i, e_j). Cette matrice est telle que :

$$i, j \in [1, n] \quad \begin{cases} s_{ij} \in [0, 1] \\ s_{ij} = s_{ji} \end{cases}$$

$$\text{et } \forall_i \qquad s_{ii} = 1$$

2.1. Algorithme

1. Choisir deux classes M et N.
2. Remplacer ces deux classes par une classe unique L. S'il ne reste qu'une classe → arrêt.
3. Recalculer s_{LK} pour les classes K restantes.
4. Aller à (1).

Au total : (n - 1) itérations.

A la $k^{ième}$ itération on obtient :

- k couples représentant les k premiers regroupements.
- k nombres qui sont les seuils associés à chaque regroupement.
- une matrice de dimension (n - k) x (n - k) des coefficients de similarité entre les (n - k) éléments qui n'ont pas été regroupés.

2.2. Exemple

Soit l'ensemble E = {1, 2, 3, 4, 5} défini par la matrice des similarités :

	1	2	3	4	5
1	1	0.95	0.80	0.30	0.50
2		1	0.60	0.25	0.30
3			1	0.40	0.35
4				1	0.55
5					1

Le premier regroupement est (1, 2)

- on remplace (1, 2) par 1 en prenant pour nouveaux coefficients $\inf(s_{1,k}, s_{2,k})$ pour k = 3, 4, 5.

	1	3	4	5
1	1	0.60	0.25	0.30
3		1	0.40	0.35
4			1	0.65
5				1

seuil 0.95
couple (1, 2)
Partition (1, 2) (3) (4) (5)

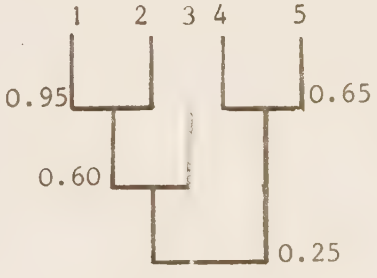
⇒

	1	3	4
1	1	0.60	0.25
3		1	0.35
4			1

seuil 0.65
couple (4, 5)
Partition (1, 2) (3) (4, 5)

	1	4
1	1	0.25
		1

seuil 0.60
couple (1, 3)
partition (1, 2, 3) (4, 5)

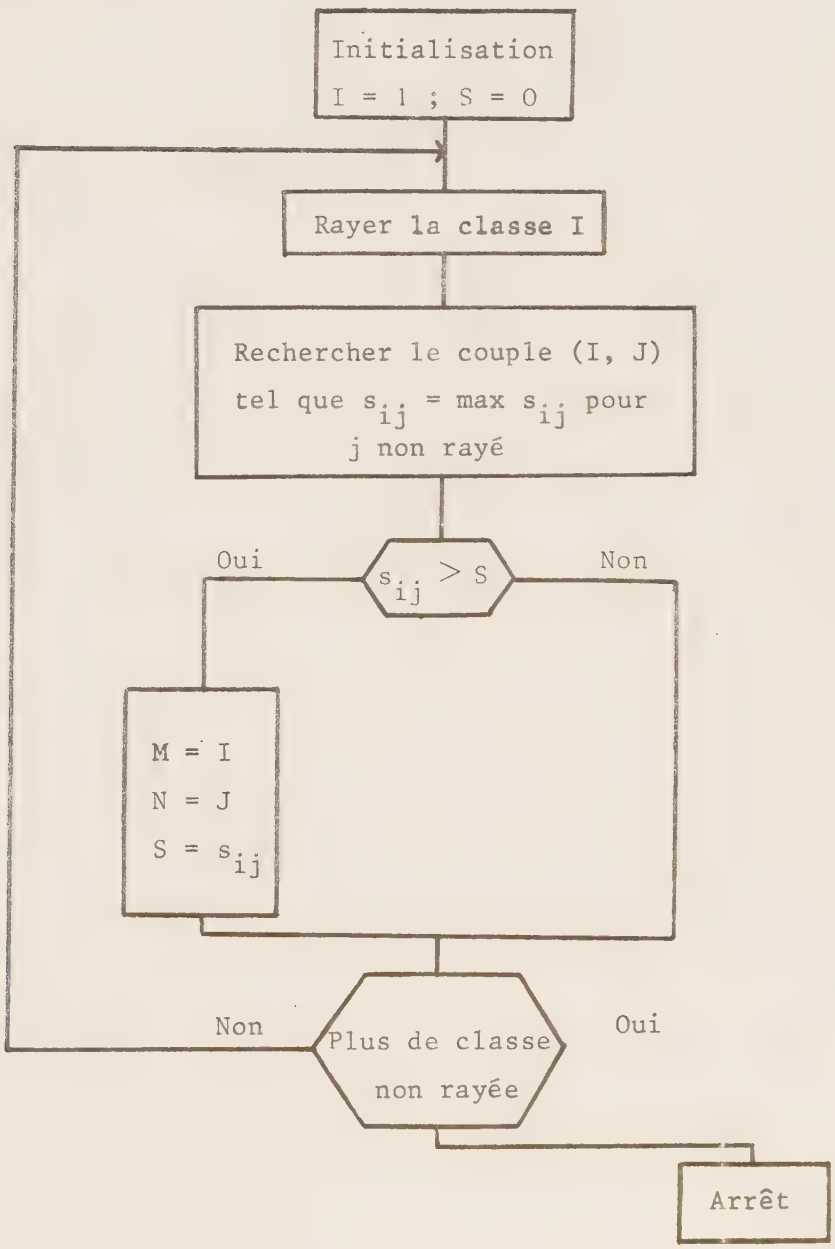


Enfin seuil 0.25
couple (1, 4)
partition (1, 2, 3, 4, 5)
d'où la hiérarchie indicée

2.3. Comment réaliser la première partie de l'algorithme

A l'étape k on dispose d'une matrice de dimension $p = n - k$.

Le couple suivant est celui qui, dans cette matrice, a le plus grand indice de similarité. On a donc à trouver l'adresse du plus grand nombre parmi $\frac{p^2 - p}{2}$ nombres.



2.4. Le choix des critères pour la définition d'une métrique

Dans le problème qui nous concerne, les éléments à comparer sont des segments de droite ou des ensembles de segments de droite. Les paramètres qui définissent un segment ne sont pas si nombreux et pourtant nous avons eu l'embarras du choix.

En effet, devions nous essayer de regrouper entre eux des éléments proches géographiquement, ou bien devions nous tenir compte d'abord des caractéristiques de forme (longueur, complexité, direction des éléments) ?

Nous avons donc été conduit à envisager plusieurs types de coefficients de similarité en gardant comme idée directrice que la construction d'une hiérarchie sur un tel ensemble pourrait nous permettre d'atteindre deux objectifs distincts :

1. Réaliser des primitives plus complexes à partir des segments élémentaires donnés. Par exemple, considérer comme primitives des groupes de plusieurs crêtes proches géographiquement les unes des autres et présentant des caractèristiques données.

2. Faciliter la recherche des éléments dans un catalogue de primitives en ne comparant les éléments qu'à des "étalons" représentant des classes de primitives (notion objective de concept).

Critères successivement utilisés

On fait les conventions suivantes :

l_i est la longueur de l'élément i

α_i est l'angle de la direction de cet élément avec l'horizontale.

d_{ij} est la distance euclidienne entre les milieux des éléments i et j

s^k_{ij} est le $k^{ième}$ indice de similarité entre i et j.

Alors :

$$s^1_{ij} = \frac{1}{1 + \dfrac{d_{ij}^2}{l_i\, l_j \cos^2(\alpha_i - \alpha_j)}}$$

$$s^2_{ij} = \frac{2 l_i\, l_j \cos(\alpha_i - \alpha_j)}{(l_i^2 + l_j^2)(1 + d_{ij}^2)}$$

$$s^3_{ij} = a_1 \frac{2\, l_i\, l_j}{l_i^2 + l_j^2} + a_2 \frac{1}{1 + d_{ij}^2} + a_3 \cos(\alpha_i - \alpha_j)$$

avec $a_1 + a_2 + a_3 = 1$

Ces trois premiers critères ont donné des résultats peu convaincants. En particulier, dans le cas de la figure (III.1) ou l'on a :

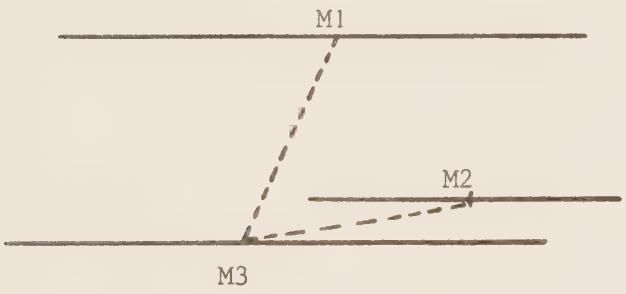


Fig. III.1

M1 M3 = M2 M3

Ces trois critères donnent

$s_{13} = s_{23}$

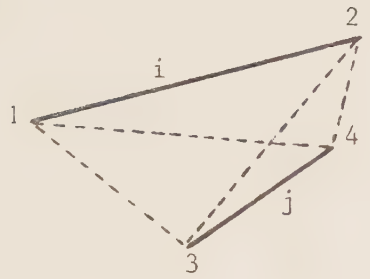
Or il peut paraître souhaitable

d'avoir $s_{23} > s_{13}$

ou encore, si $d_{ij} = 1 - s_{ij}$, on devrait avoir :

$$d_{13} > d_{23}$$

C'est ce qui nous a conduit à définir la distance suivante :



Si on désigne par 1, 2, 3, 4 les extrémités des éléments i et j de la figure ci-contre et si l_{nm} est la distance euclidienne entre les extrémités m et n on a pris :

$$d_{ij} = l_{12} + l_{34} + l_{13} + l_{24} + l_{14} + l_{23}$$

$$- (|l_{14} - l_{13}| + |l_{24} - l_{23}| + |l_{14} - l_{24}| + |l_{23} - l_{13}|)$$

On vérifie facilement que $d_{ii} = 0 \quad \forall i$

et que $\forall i \text{ et } j \qquad d_{ij} \geqslant 0$

mais d_{ij} ne vérifie pas l'inégalité triangulaire. N'importe quelle surjection décroissante de $[0, \infty[$ dans $[0, 1]$ convient pour passer de cette pseudo-distance à indice de similarité.

Par exemple, on peut prendre $s_{ij} = \frac{1}{1 + d_{ij}}$

ou encore $s_{ij} = e^{-d_{ij}}$

2.5. Résultats

Afin d'essayer la méthode nous avons assimilé les crêtes à des segments de droite et nous avons considéré une ensemble de 6 images déduites les unes des autres par de petites modifications :

- suppression d'un élément
- adjonction d'un élément
- déplacements
- coupure d'une crête en deux
- variation de la direction de certaines crêtes
- variation de la longueur de certaines crêtes.

Les planches [III.1] à [III.6] représentent ces 6 images avec les regroupements successifs en traits fins. Ces traits plus épais montrent la partition la plus stable, qui se retrouve sur les six images.

Remarque :

Une variation des conditions initiales, un changement de la métrique aboutit à une classification différente. Mais certains points sont toujours regroupés dans une même classe. Ils forment en quelque sorte le "noyau dur" de la classe. Les autres passent d'une classe à l'autre. Ce fait expérimental est à rapprocher de l'idée d'"ensemble flou" ("fuzzy sets" de ZADEH). Il serait intéressant de creuser la théorie de cette question.

Planche III.1

Planche III.2

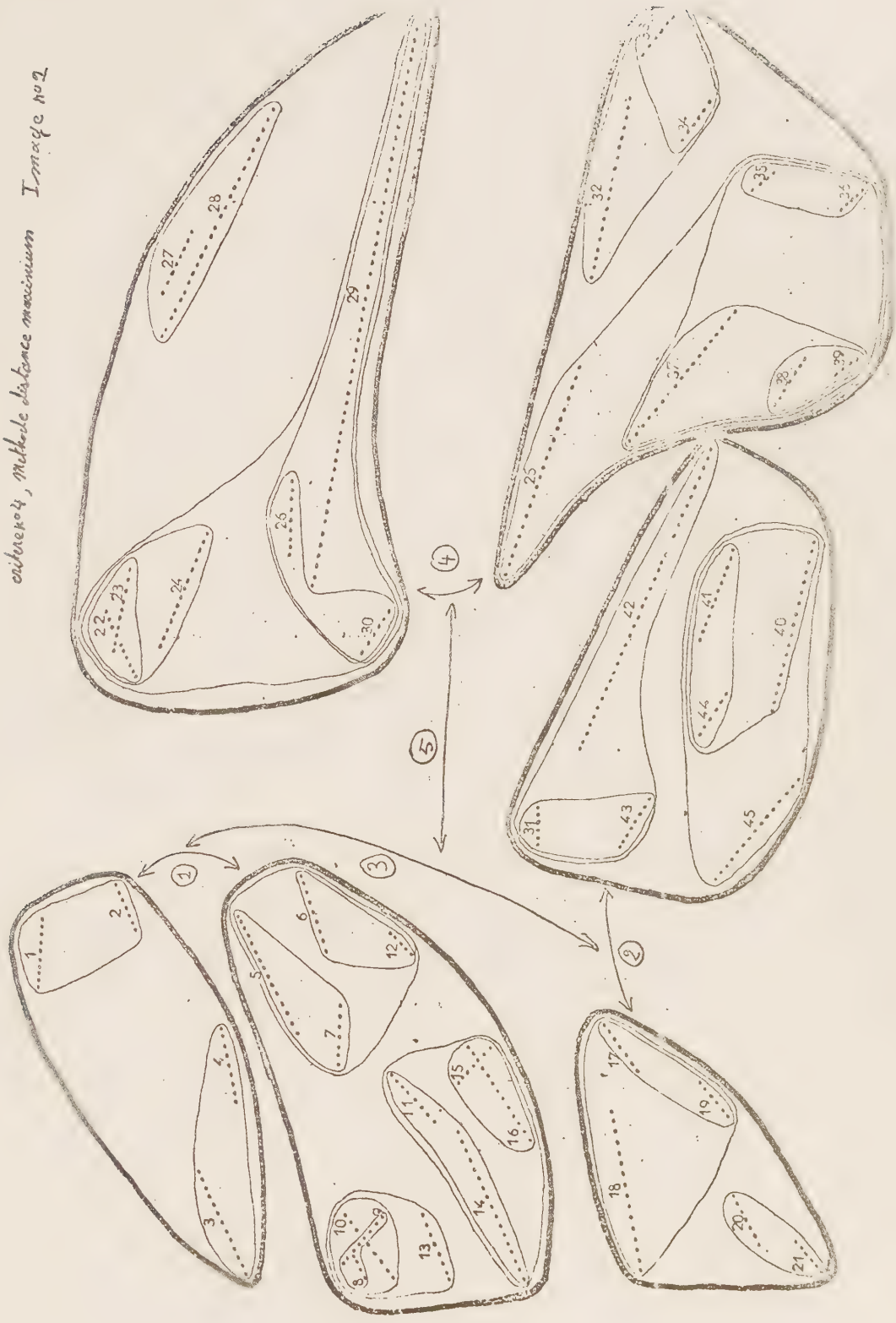


Planche III.3

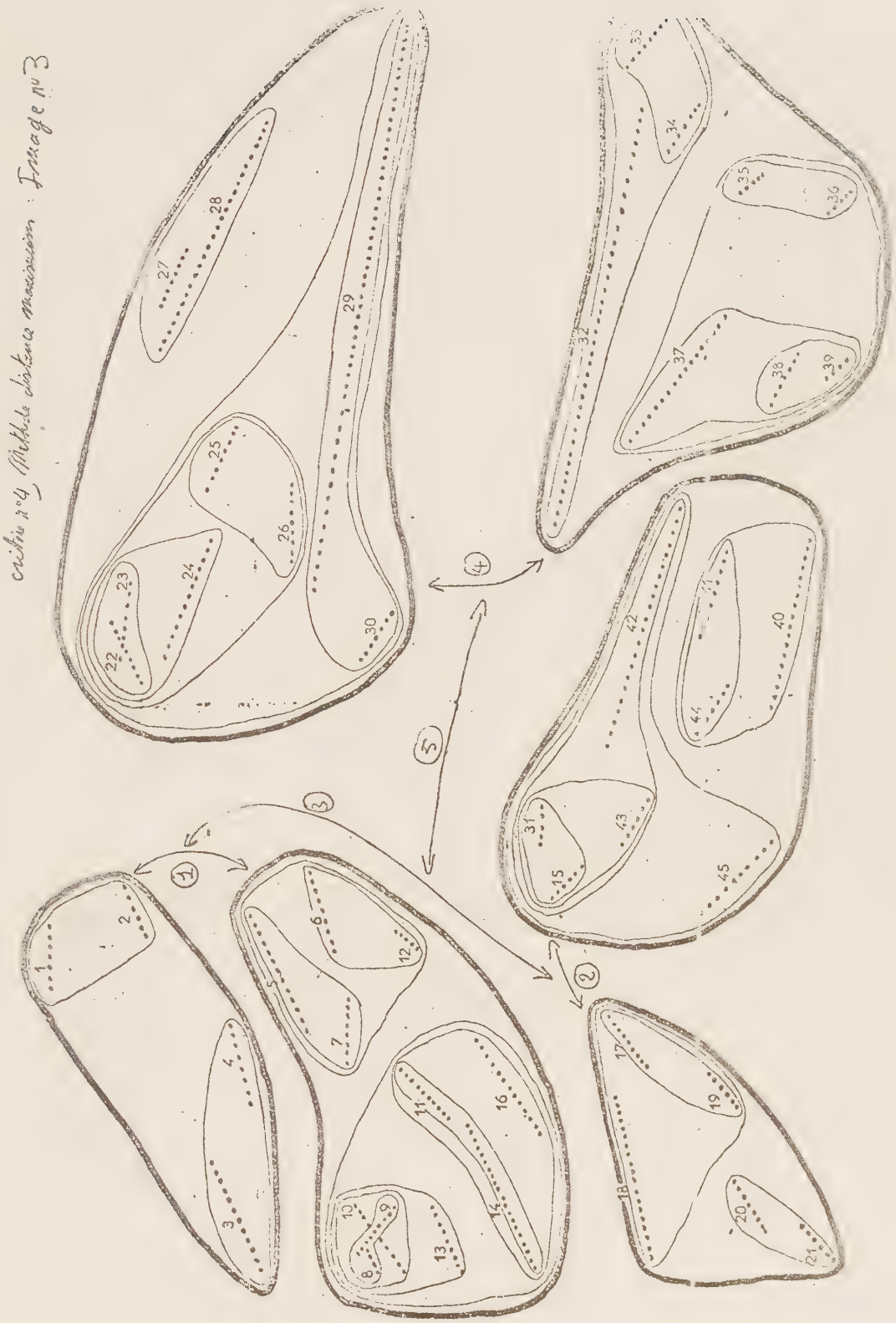


Planche III.4

Planche III.5

Planche III.6

Planche III.7.

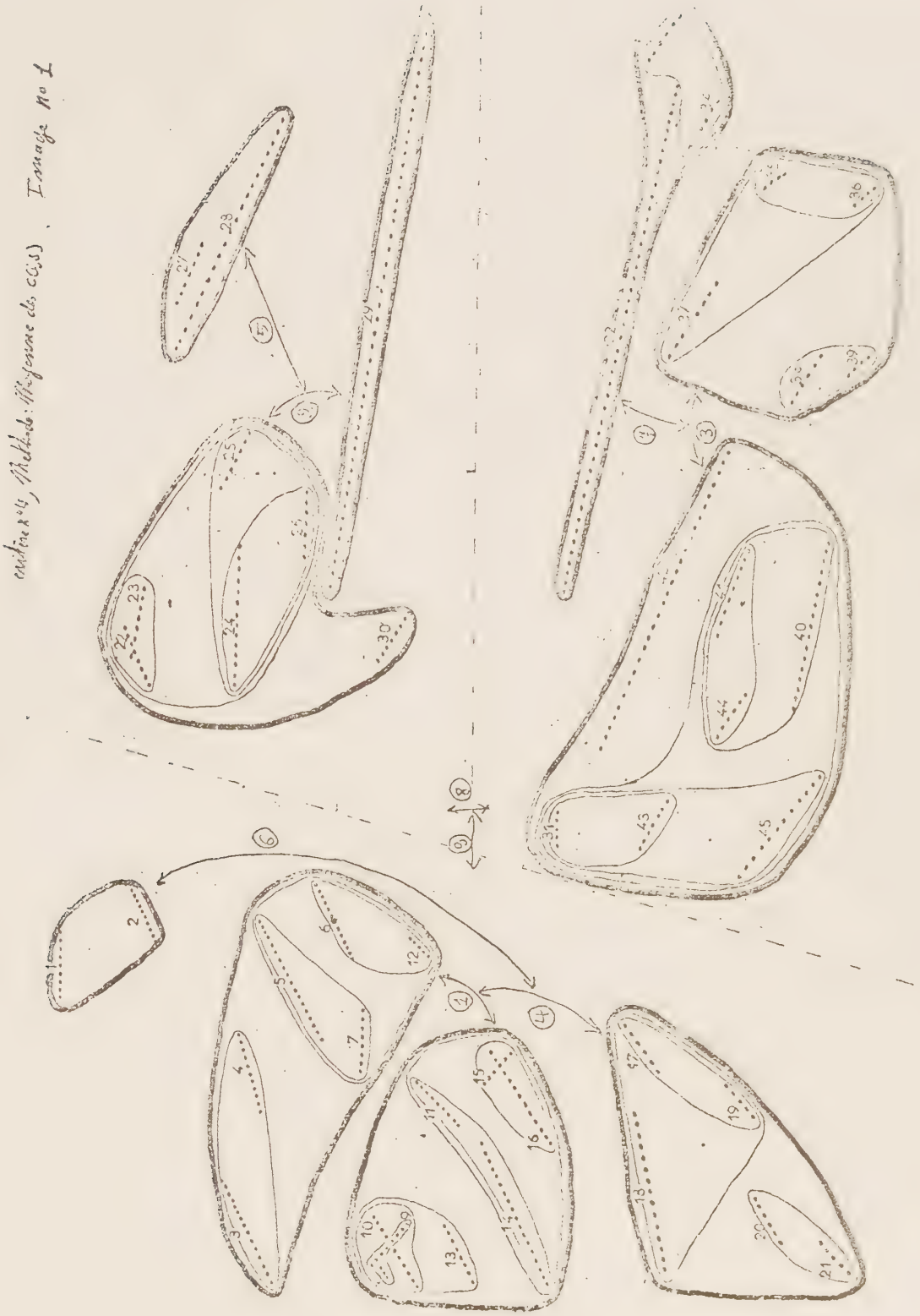


Planche III.8

Planche III.9

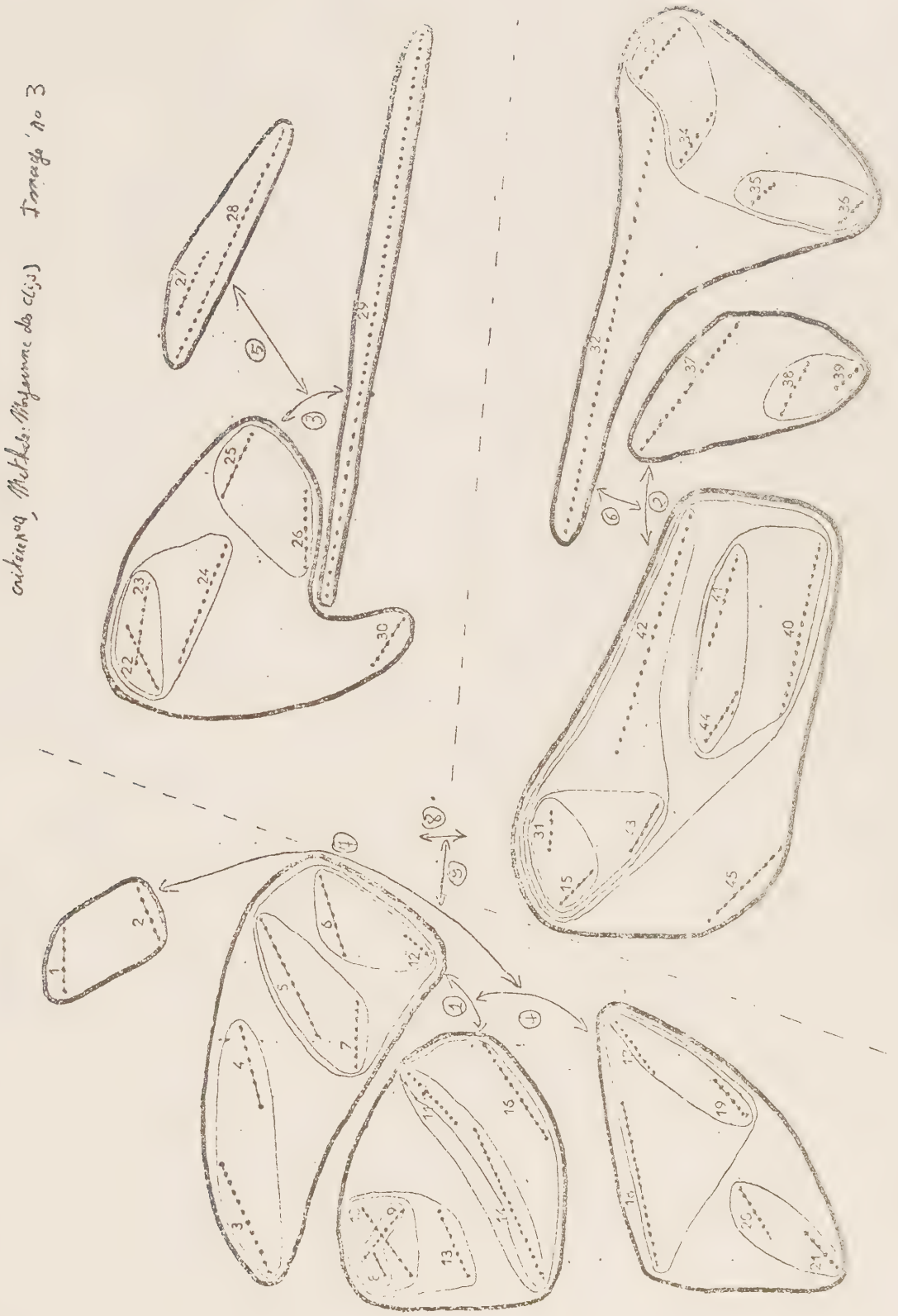


Planche III.10

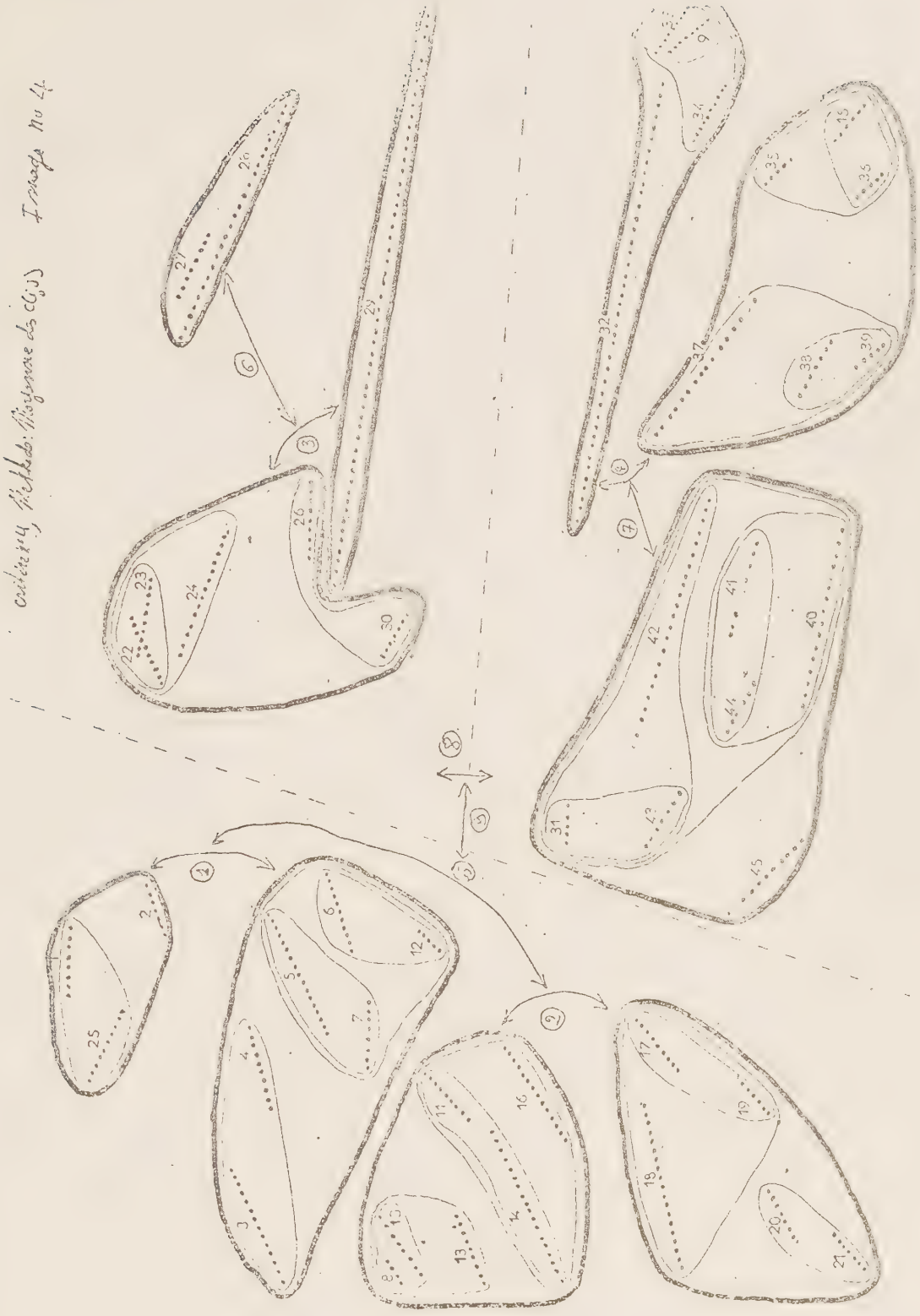


Planche III.11

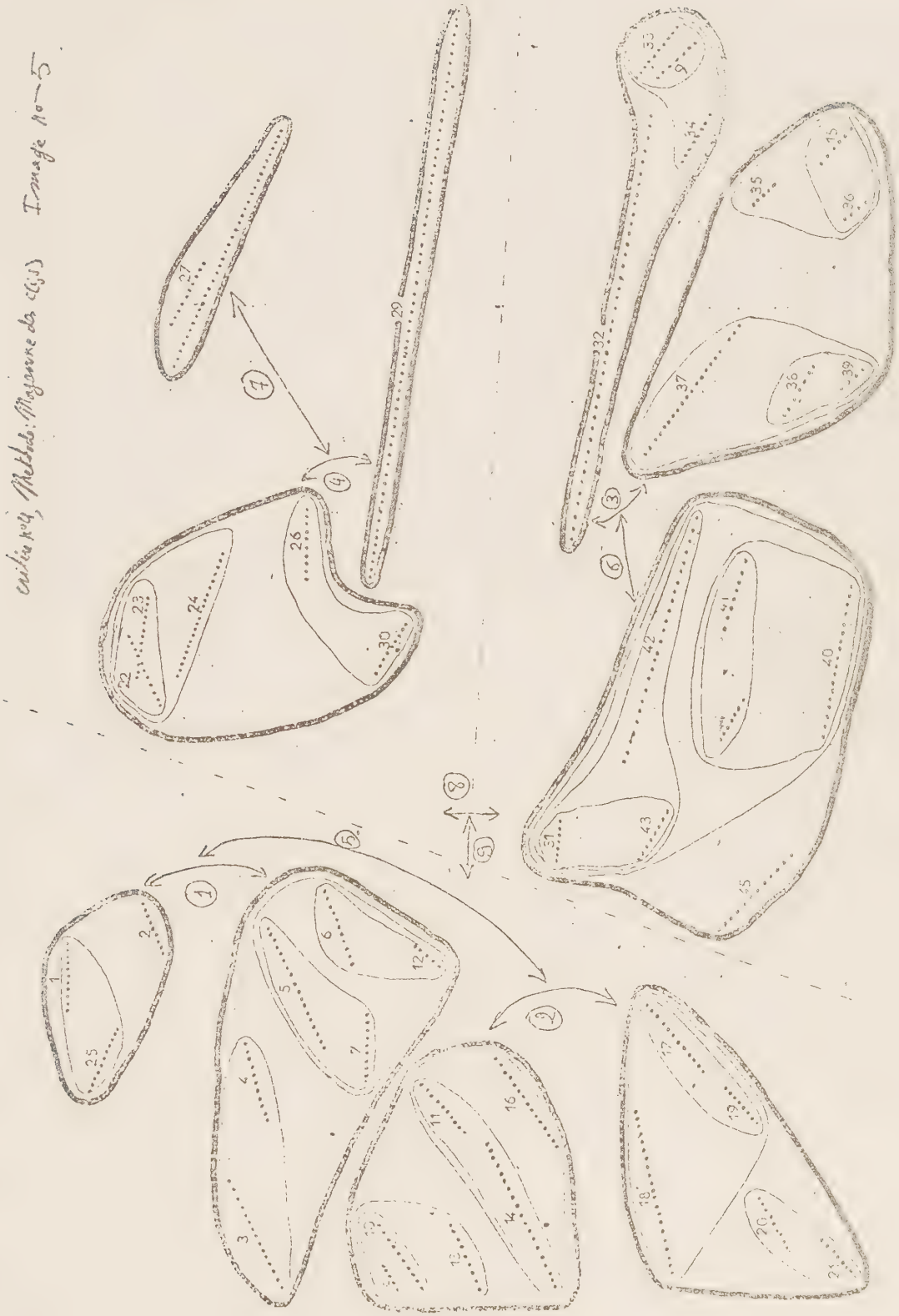
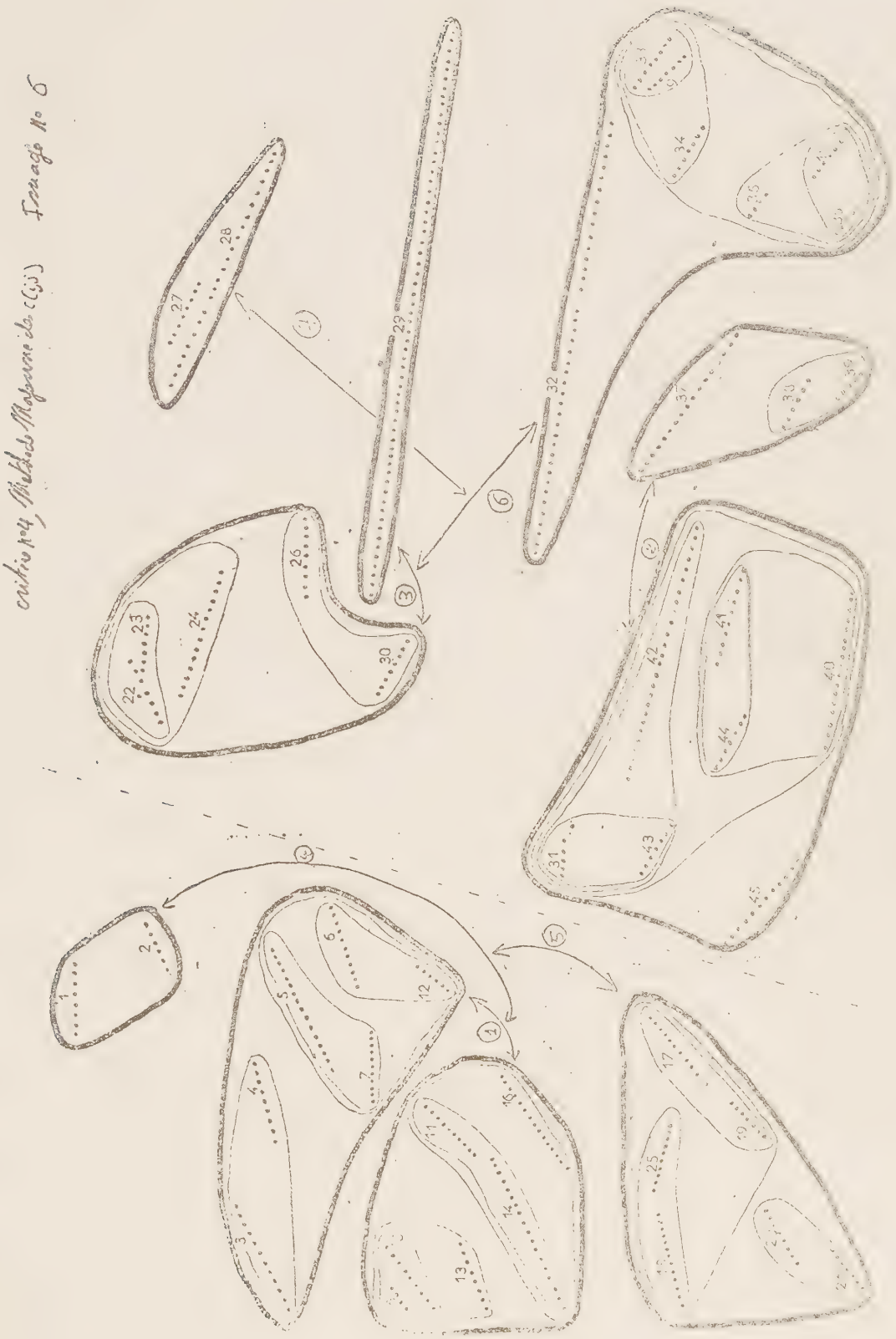


Planche III.12



Bibliographie sur le chapitre III

(1) M. ROUX : ≪ Un algorithme pour construire une hiérarchie particulière ≫ Thèse de 3ème cycle - Fac. des Sc. de Paris - Dec. 68.

(2) E. DIDAY :≪ Une nouvelle méthode en classification automatique et reconnaissance des formes : la méthode des nuées dynamiques ≫ I. R. I. A. - Document D. I. A. : 445/ED du 24/2/71.

(3) J. C. LERMAN : ≪ Les bases de la Classifiation Automatique ≫ Gauthier-Villars - 1970.

(4) J. C. SIMON, A. CHECROUN, C. ROCHE : ≪ A method of Comparing two Patterns Independant of Possible Transformations and Small Distorsions ≫ I. E. E. E. Symposium on Syntactic Pattern Recognition - CHICAGO - Oct. 1970.

(5) J. P. BENZECRI : ≪ Leçons sur l'analyse factorielle et la reconnaissance des formes ≫I. S. U. P. Paris (1966).

(6) J. P. BENZECRI : ≪ Sur l'analyse factorielle des proximités ≫ . Publications de l'I. S. U. P. Paris (1965).

(7) J. P. BENZECRI : ≪ Arbitrarité et discontinuité dans les algorithmes de classification ≫ I. S. U. P. Paris (1967).

(8) N. BOURBAKI : ≪ Livre III, chapitre 9, utilisation des réels en topologie ≫ §2, Ex. 4, Herman, éditeur, Paris.

(9) D. CORDIER : ≪ Sur 1'analyse factorielle des correspondances ≫. Thèse Rennes, 1965.

(10) J. DE LA GENIERE et F. DE LA VEGA : ≪ A propos de la classification ≫ . Publication interne n° Art/5/Juillet 66/W.F.V. Maison des Sciences de 1'Homme - rue Puiseux, Paris.

(11) C. J. JARDINE, N. JARDINE et R. SIBSON : ≪ The Structure and construction of taxonomic hierarchies, in Mathematical bioscience ≫ pp. 173-179 (1967).

(12) S. C. JOHNSON : ≪ Hierarchical clustering schemes. Psychometrika ≫ vol. 32 n°3, pp. 241-254, september 1967.

(13) P. A. DEVIJVER : ≪ A general Order Minkowski metric Pattern Classifier ≫.

C H A P I T R E IV

Analyse syntaxique des formes - Opérateurs linguistiques

§1. FORMALISATION DES OPERATIONS DE RECONNAISSANCE DES FORMES

1.1. Introduction

Comme nous l'avions annoncé dans notre chapitre d'introduction, nous allons préciser les différents opérateurs que nous avons pu mettre à jour et en faire leur étude complète.

Pour cela, nous allons supposer réalisées les différentes opérations d'analyse et traitement des images qui nous ont permis de passer de la prise de l'information brute (échantillons) à la définition des primitives de différents niveaux.

Pour un niveau N déterminé, une forme est donc définie par un ensemble X de formes primitives de nom A_i auxquelles peuvent être associées un certain nombre de valeurs numériques x_i, y_i, z_i...

$$X = \left\{ \begin{matrix} A_1 & A_2 & \cdots & A_i & \cdots & A_q \\ x_1 & x_2 & \cdots & x_i & \cdots & x_q \\ y_1 & y_2 & \cdots & y_i & \cdots & y_q \\ z_1 & z_2 & \cdots & z_i & \cdots & z_q \end{matrix} \right\} \qquad (4.1)$$

Un opérateur & appliqué à la forme X donne un résultat u_X.

Une opération de reconnaissance consiste à comparer u_X à un ensemble d'informations mises préalablement en mémoire.

Si on arrive à trouver un élément comparable à u_X on dit qu'on a extrait un élément caractéristique de niveau N + 1.

Le succès d'une opération de comparaison entre u_X et un élément en mémoire entraîne la décision que la forme X est une primitive de niveau N + 1.

Cette "primitive de niveau N + 1 " a alors un nom et des valeurs numériques associées.

La forme de départ est ainsi décrite par le nouvel ensemble $u_X = \{u_X\}$ des primitives de niveau N + 1.

Naturellement, ce processus est valable dès le premier niveau d'analyse : X est l'ensemble des échantillons obtenus par un instrument de mesure. Les opérateurs de premier niveau sont le plus souvent des opérateurs linéaires.

Par exemple, supposons que X corresponde à un signal à une dimension ; si i est le "nom" de l'un des n échantillons, si x_i la valeur attachée à cet échantillons, on appelle $\&_p$ l'opérateur dont le résultat est :

$$u_X^p = \sum_{j=1}^{j=q} \alpha_j \; x_{p-j}$$

où q et p sont des entiers tels que $q \ll n$ et $1 \leqslant q \leqslant n - q$.
C'est l'ensemble des q paramètres $\{\alpha_j\}$ qui définit l'opérateur $\&_p$. Le résultat u_X^p est une valeur numérique obtenue par une translation de p unités et une multiplication scalaire. La décision d'avoir reconnu ou non un élément est fonction de cette valeur prise par u_X^p. En général, si u_X^p dépasse un certain seuil u_0, l'élément correspondant à l'opérateur $\&_p$ est reconnu.

Les filtres digitaux et les perceptrons utilisent de tels opérateurs linéaires. Il est facile d'étendre le procédé aux images c'est-à-dire aux signaux à deux dimensions, (ref.).

Une forme correspond le plus souvent à l'examen d'un objet concret, qui peut subir différentes transformations géométriques devant l'appareil de mesure : translation, rotation, éloignement, etc.... On souhaite bien entendu, que la chaîne des opérateurs utilisés pour reconnaître l'objet donne un résultat invariant en fonction de ces transformations. C'est bien la réaction d'un homme examinant une scène réelle. En quelque sorte le résultat cherché est une représentation invariante.

Il est donc du plus grand intérêt de trouver des opérateurs invariants en fonction des transformations géométriques usuelles. Il est clair que les opérateurs linéaires envisagés ne sauraient avoir cette qualité. Les opérateurs de regroupement sont invariants en fonction des translations, des homothéties et souvent de la rotation d'une figure dans son plan. En fait, nous avons cherché à dégager des opérateurs invariants de type nouveau, intermédiaires entre ces opérateurs de regroupement et les opérateurs syntaxiques. Nous les appelons opérateurs linguistiques ; nous croyons qu'ils représentent la contribution la plus originale de ce travail.

1.2. Opérateurs linguistiques

Bon nombre d'auteurs tels que EDEN [4], MINSKY [8], NARASHIMAN [9], KIRSH [7], ont évoqué des processus de "description d'images", commençant ainsi une approche syntaxique des problèmes de reconnaissance des formes plus complexes. Citons les remarques de FU et SWAIN [6] :

≪ La Reconnaissance de formes syntaxiques est un essai pour adapter la réunion ≫
≪ des langages formels, qui procurent à la fois une notation (grammaire) et ≫
≪ un mécanisme d'analyse (syntaxique) au problème de la représentation et de ≫
≪ l'analyse des formes... L'analyse syntaxique ne peut être appliquée que si ≫
≪ un modèle grammatical pour la génération des données et/ou le processus ≫
≪ d'analyse peut être formulé. (Ceci n'implique pas nécessairement que les ≫
≪ données soient engendrées concrètement par un mécanisme qui opère de la ≫
≪ même manière que le modèle). Un problème pratique connexe est le dévelop- ≫
≪ pement de procèdures efficace d'analyses basées sur le modèle grammatical. ≫
≪ La littérature concernant la reconnaissance syntaxique de formes donne ≫
≪ presque exclusivement des modèles grammaticaux, bien que le développement ≫
≪ d'algorithmes d'analyse reçoive de plus en plus d'attention... ≫.

En fait, ainsi que le reconnaît KIRSCH, les techniques d'analyse syntaxique ne réussissent que si la grammaire générative est complètement connue. Mais comme pour les langages naturels, on peut se demander si, pour les images, il existe toujours des grammaires du type de "CHOMSKY".

Finalement, nous avons voulu retenir deux idées qui nous paraissent primordiales :

- ne pas interpréter les résultats obtenus comme des quantités numériques obtenues mais considérer ces éléments comme les mots d'un langage. (Des valeurs numériques peuvent leur être associées).
- essayer de trouver des "règles valables pour des classes de formes les plus grandes possibles (notion d'invariance).

Par conséquent, utilisant le formalisme du §1 , on peut proposer des opérateurs tels que :

- le résultat u_X est constitué par un ou plusieurs mots construits sur l'alphabet des formes primitives. Pour cela, il suffit de trouver l'algorithme approprié et il n'est point nécessaire d'une grammaire générative. Cet algorithme ou "opérateur d'analyse" est exécutable bien entendu par un ordinateur.
- le résultat u_X est invariant par rapport à un certain nombre de transformations de l'ensemble X. C'est en fait, cette recherche de l'invariance qui nous a amené à la notion d'opérateur linguistique.

Remarque : une forme est constituée par N primitives

$$X = \{A_1 \quad A_2 \ \ldots \ A_i \ \ldots \ A_N\}$$

Chaque A_i est représenté dans un espace $\mathbb{R}^n$ où n est le nombre de paramètres attachés à chaque primitive. Au chapitre précédent, nous nous sommes toujours ramené à une représentation dans $\mathbb{R}^2$.

Dans ce qui suit, les opérateurs seront appliqués à des ensembles de points de $\mathbb{R}^2$. Il est clair qu'ils pourraient opérer dans $\mathbb{R}^n$.

La décision de reconnaissance est prise en comparant le résultat obtenu u_X avec des résultats déjà enregistrés en mémoire.

§2. UN OPERATEUR INDEPENDANT DES TRANSFORMATIONS GEOMETRIQUES SIMPLES ET DES PETITES TRANSFORMATIONS

2.1. Définition

Les travaux de SHEPARD [1] et BENZECRI [2] nous ont donné l'idée d'utiliser les propriétés des "ordonnances" pour résoudre le problème suivant :

Supposons qu'une forme soit constituée par N primitives et que l'on représente ces N primitives par N points de $\mathbb{R}^2$.

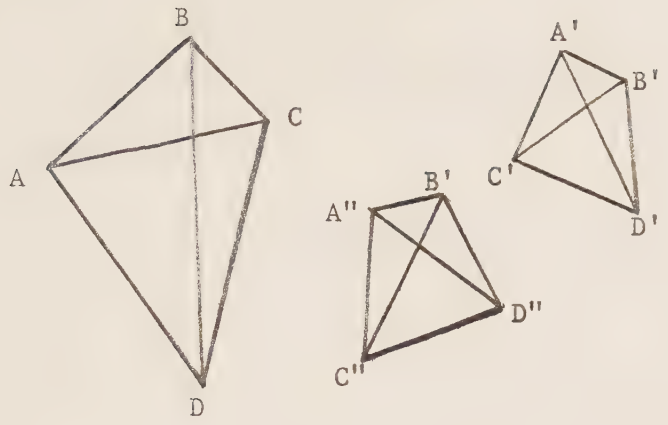
Les $\frac{N(N-1)}{2}$ distances euclidiennes entre ces points peuvent être ordonnées (par exemple dans l'ordre des valeurs décroissantes). Appelons ij le segment qui joint le point de nom i au point de nom j, et d_{ij} la distance euclidienne correspondante. Il est alors possible d'établir une correspondance entre la suite décroissante $\{d_{ij}\}$ et l'ensemble des segments {ij}.

Ce dernier ensemble, ainsi ordonné, est appelé une "ordonnance" sur {i} (cf. [2]).

On peut alors énoncer la propriété suivante :

Si deux ensembles {i} et {i'} sont tels que l'ordonnance {ij} déduite du premier ensemble, correspond terme à terme à l'ordonnance {i'j'} obtenue à partir du second ensemble, alors il est possible de trouver une transformation géométrique simple faisant passer de {i} à {i'} et telle que $\text{Max } d_{ii'} < \epsilon$, avec ϵ petit par rapport aux $\{d_{ij}\}$.
Dans ce cas nous dirons que ces deux ensembles sont strictement similaires.

Mais il est encore plus intéressant de se demander si, deux ensembles {i} et {i'} étant similaires, les ordonnances {ij} et {i'j'} qui peuvent en être déduites, se correspondent terme à terme ? Et dans la négative, quelles opérations peuvent rendre cette correspondance possible ? Par exemple, considérons les trois ensembles F, F', F" de la figure 3.1, considérés comme similaires avec $\epsilon = 0.1$.



F	F'	F"
BD	A'D'	A"D"
CD	B'D'	B"C"
AC	B'C'	B"D"
AB	A'C'	C" D"
AD	C'D'	A"C"
BC	A'B'	A"B"

Figure 3.1

On trouve facilement la correspondance entre les figures

$$F = \{i\} \quad \text{et} \quad F' = \{i'\}$$

$$A' \to B \; ; \quad B' \to C \; ; \quad C' \to A \; ; \quad D' \to D$$

Mais il est impossible de trouver une telle correspondance entre les points de F et ceux de F", bien que ces deux figures soient similaires au sens défini plus haut. Essayons d'appliquer à F" la même transformation qui fait passer de F' à F. Nous obtenons la suite :

$$F'' \to BD, AC, CD, AD, AB, BC$$

alors que nous avions :

$$F \to BD, CD, AC, AB, AD, BC$$

On voit que les deux suites diffèrent par deux permutations :

(AC, CD) d'une part ; (AB, AD) d'autre part

C'est ainsi que nous avons pensé à la méthode de comparaison qui suit.

2.2. Le "parenthésage" et le principe d'affectation

Le parenthésage

Admettons une imprecision e sur l'évaluation des distances de la figure. Deux segments $i_1\ j_1$ et $i_2\ j_2$ de cette figure, tels que $|d_{i_1\ j_{i_1}} - d_{i_2\ j_{i_2}}| < e$ peuvent permuter dans l'ordonnance {ij}. L'ensemble des permutations de ce type définit une classe de "figures similaires" ; la classe $\mathcal{F}$ des formes similaires à la figure F = {i}. Nous appelons parenthésage la méthode de représentation de cette classe, c'est-à-dire l'opération qui consiste à trouver et repérer les éléments de l'ordonnance qui sont susceptibles de permuter.

Si deux éléments peuvent permuter, tous les éléments intermédiaires dans l'ordonnance peuvent permuter entre eux et avec les éléments extrêmes ce sous-ensemble d'éléments permutables est délimité par des parenthéses.

Exemple : reprenons la figure F et considérons que l'évaluation des distances euclidiennes de cette figure ait été faite avec une erreur e. Alors la classe des figures similaires à F avec une précision e sera définie par la suite {ij} munie de parenthèses :

$$F \rightarrow BD, (CD, AC, (AB), AD, BC)$$

Règle : pour obtenir les suites {i' j'}, correspondant à toutes les figures possibles similaires à une figure F, il faut effectuer toutes les permutations possibles sur les segments ij qui dans l'ordonnance se trouve entre une parenthèse ouvrante et la parenthèse fermante suivante.

Le principe d'affectation

Considérons deux figures F et F' telles que l'une, F, soit un catalogue (figure de référence) et l'autre F' soit une donnée à identifier.

Si F est représentée par l'ordonnance parenthèsée suivante :

$$F \rightarrow BD, (CD, AC, (AB), AD), BC$$

Cela signifie, par exemple, que CD, AC, AB peuvent permuter entre eux, mais AC et AD ne le peuvent pas.

Représentons F et F' avec leurs ordonnances parallèles (fig. 3.2).

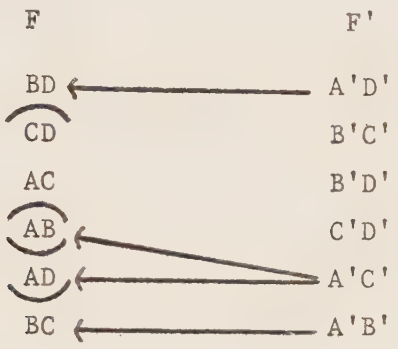


Figure 3.2

En affectant A' sans tenir compte du parenthésage, nous trouvons les possibilités suivantes :

A' → B ou D (ligne 1)
A' → A ou D (ligne 5)
A' → B ou C (ligne 6)

L'affectation est impossible puisqu'il y a contradiction entre les trois hypothèses possibles.

Par contre, en tenant compte des parenthèses, c'est-à-dire des permutations possibles, nous trouvons que :

A' → B ou D (ligne 1)
A' → A, B ou D (ligne 5)
A' → B ou C (ligne 6)

Dans ces trois hypothèses, une même proposition se retrouve :

$$\boxed{A' \rightarrow B}$$

On voit finalement que la classe $\mathcal{F}$ est d'autant plus grande que l'imprécision $\underline{e}$ permise sur l'évaluation des distances est grande ; d'où une plus grande possibilité d'affectation de la figure F' à cette classe.

2.3. Méthode de comparaison de deux formes

2.3.1. Représentation matricielle des affectations.

Considérons deux figures de 4 points :

$$F = \{A\ B\ C\ D\} \quad \text{et} \quad F' = \{A'\ B'\ C'\ D'\}$$

Nous pouvons représenter l'affectation point à point de ces deux formes à l'aide d'une matrice dont l'élément courant $e_{ij} = 1$ si le point i de F est affecté au point j de F'.
$e_{ij} = 0$ dans le cas où i ne peut pas être affecté à j.

	A	B	C	D
A'	0	1	0	0
B'	0	1	1	0
C'	1	1	0	1
D'	1	1	0	1

Figure IV.1

Naturellement, a priori tous les éléments valent 1. Lorsqu'on examine les "ordonnances" correspondant à F et F', il apparaît des impossibilités et plusieurs zéros remplacent des 1 dans la matrice. Par exemple, on obtient la matrice de la figure IV.1. Dans une telle matrice, l'affectation est totale si chaque ligne et chaque colonne renferme un 1 et un seul.

2.3.2. Première réduction de la matrice

Lorsqu'une ligne (ou une colonne) ne renferme qu'un seul 1, on peut déduire que l'affectation du point correspondant est réalisée, et par suite annuler tous les centres éventuels de la colonne (ou la ligne) perpendiculaire. Ainsi, dans la matrice de la figure IV.1 le point A' ne peut être affecté qu'au point B et le point C ne peut être affecté qu'à B'.

La nouvelle matrice obtenue est donnée par la figure IV.2.

	A	B	C	D
A'	0	1	0	0
B'	0	0	1	0
C'	1	0	0	1
D'	1	0	0	1

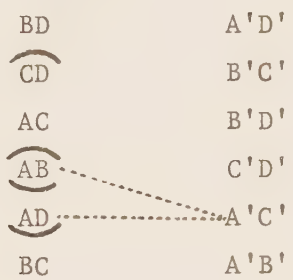
	A	B	C	D
A'	0	1	0	0
B'	0	0	1	0
C'	1	0	0	0
D'	0	0	0	1

Figure IV.2

L'affectation n'est pas totale et on peut essayer de réduire à nouveau la matrice.

2.3.3. Deuxième réduction

Si nous reprenons les ordonnances correspondant à F et F' :



nous constatons que A'C' est en face de la parenthèse (AB, AD) ; or A est affecté à B donc si une affectation totale est possible ce n'est qu'avec C' affecté à A.

Autrement dit, si un point i' de F' a été affecté à un point i de F, alors pour tout segment i'j' de F', j' ne peut être affecté qu'aux points j qui sont couplés avec i dans l'ordonnance et tels que ij puisse venir en face de i'j' après une éventuelle permutation autorisée. Dans l'exemple précédent, la matrice d'affectation devient celle de la figure IV.4 et l'affectation est totale.

Bien sûr pour des formes constituées par un nombre de primitives plus grand on n'aboutit pas aussi rapidement à une affectation totale. On peut alors appliquer successivement et plusieurs fois ces deux types de réduction jusqu'à ce que la matrice d'affectation ne varie plus.

Trois cas sont alors possibles :

1. L'une au moins des lignes ou des colonnes ne comporte que des zéros : l'affectation est impossible.
2. Chaque ligne et chaque colonne au moins ne comporte qu'un seul 1 : l'affectation est réalisée.
3. Une ligne ou une colonne au moins comporte plusieurs 1 : l'affectation est indéterminée.

Cette dernière éventualité est représentée par le cas de la figure IV.5 : il n'est pas possible de trouver d'autre réduction. Plusieurs affectations sont alors possibles.

1	0	0	0
0	1	0	0
0	0	1	1
0	0	1	1

Figure IV.5

2.4. Description des expériences et résultats

Nous avons voulu réaliser une simulation pour déterminer la validité de cette méthode. Pour cela nous avons généré un certain nombre de figures de manière aléatoire et nous les avons comparées deux à deux à l'aide de ces programmes.

2.4.1. La génération aléatoire des figures

Une première figure F1 de 10 points est réalisée en prenant, à l'aide d'un programme de génération de nombres aléatoires distribués uniformément, les 20 coordonnées des points de la figure (planche IV.1). Deux autres figures F2 et F3 sont construites alors en supposant que chacun de leurs points est pris au hasard dans un cercle de rayon ϵ autour du point correspondant de F1. Ce qui revient à ajouter une valeur aléatoire $\delta \in [0, 2\epsilon]$ à chacune des coordonnées $(x - \epsilon)$ et $(y - \epsilon)$ calculées précédemment.

L'ensemble F1, F2, F3 des trois formes ainsi engendrées constitue une "classe de similarité".

Nous avons construit trois telles classes, c'est-à-dire au total 9 figures F1 ... à ... F9 et nous leur avons appliqué les programmes de comparaison deux à deux.

2.4.2. Les différentes étapes des comparaisons

- pour chaque figure nous construisons l'ordonnance de ses différents "segments".
- chaque ordonnance est comparée à toutes les autres successivement. Les parenthèses sont placées en admettant une erreur "e" sur les distances euclidiennes.
- pour chaque comparaison nous faisons varier e jusqu'à ce que l'affectation soit possible ou indéterminée. Si nous appelons e_2 la valeur de e pour laquelle l'affectation devient possible et e_3 celle pour laquelle l'affectation devient indéterminée, nous obtenons le schéma général suivant :

Quelquefois on passe directement de l'impossibilité à l'indétermination ($e_2 = e_3$). Alors, on est sûr que les deux figures "ne se ressemblent pas".

Autrement, on dit que les figures sont "comparables avec la précision e_2".

Remarque : nous retrouvons ainsi le point de vue déjà rencontré dans la représentation d'une surface à l'aide d'un système informatique. Les décisions ne peuvent être prises qu'avec une précision déterminée. Inversement, à l'intérieur d'une certaine précision toutes les représentations sont absolument indiscernables. Voilà qui nous sépare de la position "mathématique" habituelle.

Par exemple dans la "Valeur de la Science" Poincaré examine la relation si A = B et B = C alors A = C qui conduit à des paradoxes de représentation. Il n'en est rien pour nous... La notion d'"égalité" ou plutôt d'"équivalence" n'est pas transitive dans les procédés décrits.

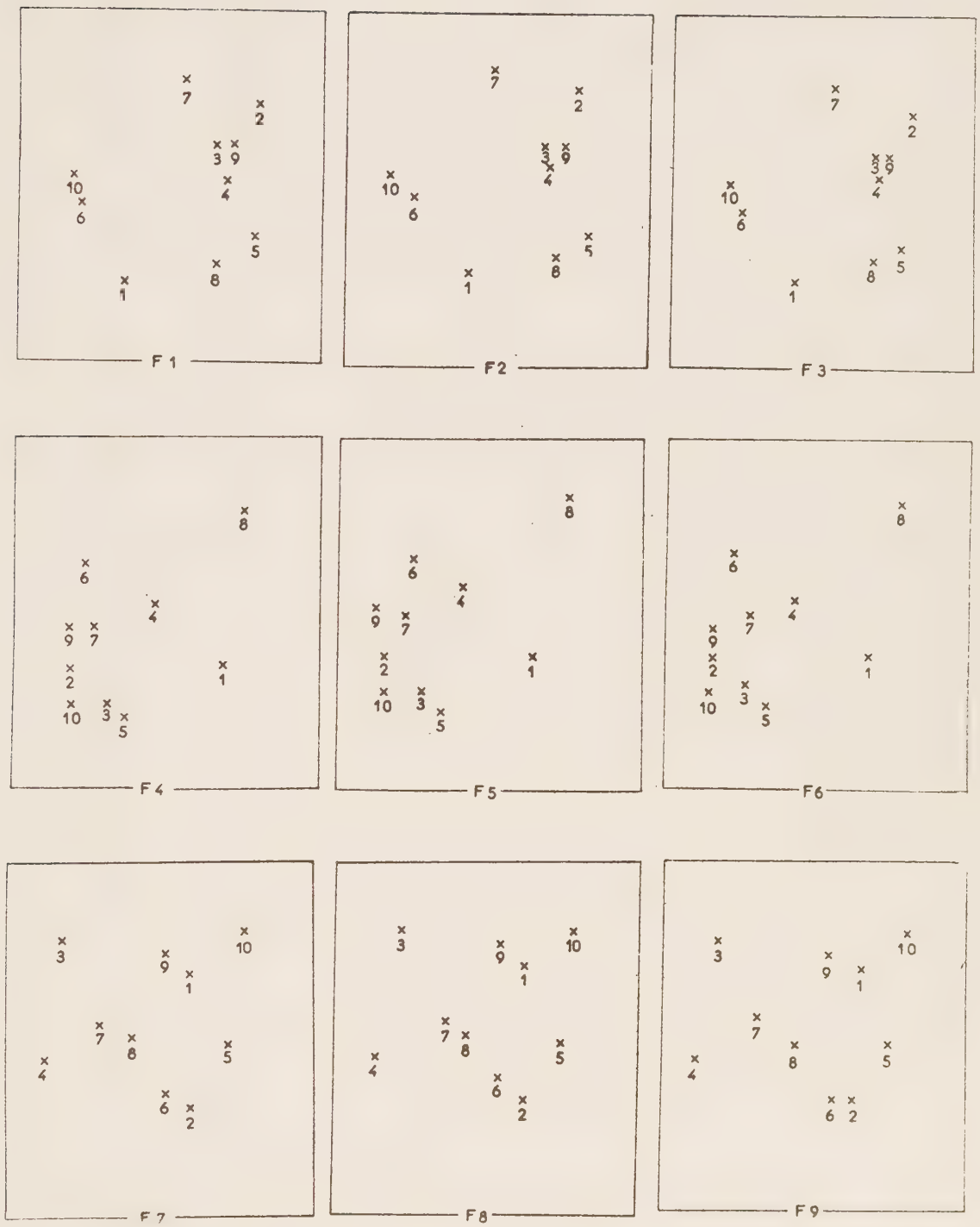
2.4.3. Confirmations expérimentales

Les expériences ont été faites avec des valeurs de ϵ allant de 0.1 à 0.3 (c'est-à-dire des déformations possibles de 10 à 30 % de la dimension moyenne de la figure). Nous publions planches IV.10 et IV.11 les résultats correspondant à $\epsilon = 0.1$.

Dans la liste de la planche IV.10, chaque ligne représente les noms i et j des figures comparées, la valeur P2 à partir de laquelle l'affectation est possible ou indéterminée, et les noms des points affectés (0 signifie que l'affectation du point correspondant est indéterminée).

A travers ces résultats on peut voir que les trois classes primitivement créées sont retrouvées. Deux figures appartenant à des classes différentes conduisent à une indétermination et la valeur de e_2 est alors élevée.

La planche IV.11 montre les e_{ij}, seuils à partir desquels la figure i peut être affectée à la figure j de façon unique ou indéterminée.



Patterns corresponding to Tables 1 and 2.

Les pages 90 à 97 ont été supprimées afin d'améliorer la présentation des planches III.1 à III.9.

COUPLE	1, 2	P2=	.0312	1	2	3	4	5	6	7	8	9	10	AFFEC. REA.
COUPLE	1, 3	P2=	.0562	1	2	3	4	5	6	7	8	9	10	AFFEC. REA.
COUPLE	1, 4	P2=	.3250	0	0	0	0	0	0	0	0	0	0	
COUPLE	1, 5	P2=	.3625	0	0	0	0	0	0	0	0	0	0	
COUPLE	1, 6	P2=	.3812	0	0	0	0	0	0	0	0	0	0	
COUPLE	1, 7	P2=	.1250	0	0	0	0	0	0	0	0	0	0	
COUPLE	1, 8	P2=	.1375	0	0	0	0	0	0	0	0	0	0	
COUPLE	1, 9	P2=	.1312	0	0	0	0	0	0	0	0	0	0	
COUPLE	2, 1	P2=	.0375	1	2	3	4	5	6	7	8	9	10	AFFEC. REA.
COUPLE	2, 3	P2=	.0437	1	2	3	4	5	6	7	8	9	10	AFFEC. REA.
COUPLE	2, 4	P2=	.3187	0	0	0	0	0	0	0	0	0	0	
COUPLE	2, 5	P2=	.3562	0	0	0	0	0	0	0	0	0	0	
COUPLE	2, 6	P2=	.3500	0	0	0	0	0	0	0	0	0	0	
COUPLE	2, 7	P2=	.1250	0	0	0	0	0	0	0	0	0	0	
COUPLE	2, 8	P2=	.1375	0	0	0	0	0	0	0	0	0	0	
COUPLE	2, 9	P2=	.1312	0	0	0	0	0	0	0	0	0	0	
COUPLE	3, 1	P2=	.0562	1	2	3	4	5	6	7	8	9	10	AFFEC. REA.
COUPLE	3, 2	P2=	.0437	1	2	3	4	5	6	7	8	9	10	AFFEC. REA.
COUPLE	3, 4	P2=	.3187	0	0	0	0	0	0	0	0	0	0	
COUPLE	3, 5	P2=	.3562	0	0	0	0	0	0	0	0	0	0	
COUPLE	3, 6	P2=	.3500	0	0	0	0	0	0	0	0	0	0	
COUPLE	3, 7	P2=	.1250	0	0	0	0	0	0	0	0	0	0	
COUPLE	3, 8	P2=	.1375	0	0	0	0	0	0	0	0	0	0	
COUPLE	3, 9	P2=	.1312	0	0	0	0	0	0	0	0	0	0	
COUPLE	4, 1	P2=	.1375	0	0	0	0	0	0	0	0	0	0	
COUPLE	4, 2	P2=	.1125	0	0	0	0	0	0	0	0	0	0	
COUPLE	4, 3	P2=	.1250	0	0	0	0	0	0	0	0	0	0	
COUPLE	4, 5	P2=	.0562	1	2	3	4	5	6	7	8	9	10	AFFEC. REA.
COUPLE	4, 6	P2=	.0625	1	2	3	4	5	6	7	8	9	10	AFFEC. REA.
COUPLE	4, 7	P2=	.1437	0	0	0	0	0	0	0	0	0	0	
COUPLE	4, 8	P2=	.1625	0	0	0	0	0	0	0	0	0	0	
COUPLE	4, 9	P2=	.1375	0	0	0	0	0	0	0	0	0	0	
COUPLE	5, 1	P2=	.1375	0	0	0	0	0	0	0	0	0	0	
COUPLE	5, 2	P2=	.1250	0	0	0	0	0	0	0	0	0	0	
COUPLE	5, 3	P2=	.1250	0	0	0	0	0	0	0	0	0	0	
COUPLE	5, 4	P2=	.0437	1	2	3	4	5	6	7	8	9	10	AFFEC. REA.
COUPLE	5, 6	P2=	.0625	1	2	3	4	5	6	7	8	9	10	AFFEC. REA.
COUPLE	5, 7	P2=	.1437	0	0	0	0	0	0	0	0	0	0	
COUPLE	5, 8	P2=	.1625	0	0	0	0	0	0	0	0	0	0	
COUPLE	5, 9	P2=	.1375	0	0	0	0	0	0	0	0	0	0	
COUPLE	6, 1	P2=	.1312	0	0	0	0	0	0	0	0	0	0	
COUPLE	6, 2	P2=	.1250	0	0	0	0	0	0	0	0	0	0	
COUPLE	6, 3	P2=	.1250	0	0	0	0	0	0	0	0	0	0	
COUPLE	6, 4	P2=	.0625	1	2	3	4	5	6	7	8	9	10	AFFEC. REA.
COUPLE	6, 5	P2=	.0812	1	2	3	4	5	6	7	8	9	10	AFFEC. REA.
COUPLE	6, 7	P2=	.1437	0	0	0	0	0	0	0	0	0	0	
COUPLE	6, 8	P2=	.1500	0	0	0	0	0	0	0	0	0	0	
COUPLE	6, 9	P2=	.1187	0	0	0	0	0	0	0	0	0	0	
COUPLE	7, 1	P2=	.1562	0	0	0	0	0	0	0	0	0	0	
COUPLE	7, 2	P2=	.1500	0	0	0	0	0	0	0	0	0	0	
COUPLE	7, 3	P2=	.1500	0	0	0	0	0	0	0	0	0	0	
COUPLE	7, 4	P2=	.3937	0	0	0	0	0	0	0	0	0	0	

Table 6 : (beginning)

COUPLE	7,5	P 2 =	.4187	0	0	0	0	0	0	0	0	0	0	
COUPLE	7, 6	P2=	.4375	0	0	0	0	0	0	0	0	0	0	
COUPLE	7, 8	P2=	.0500	1	2	3	4	5	6	7	8	9	10	AFFEC. REA.
COUPLE	7, 9	P2=	.0437	1	2	3	4	5	6	7	8	9	10	AFFEC. REA.
COUPLE	8, 1	P2=	.1562	0	0	0	0	0	0	0	0	0	0	
COUPLE	8, 2	P2=	.1500	0	0	0	0	0	0	0	0	0	0	
COUPLE	8, 3	P2=	.1562	0	0	0	0	0	0	0	0	0	0	
COUPLE	8, 4	P2=	.3937	0	0	0	0	0	0	0	0	0	0	
COUPLE	8, 5	P2=	.4187	0	0	0	0	0	0	0	0	0	0	
COUPLE	8, 6	P2=	.4375	0	0	0	0	0	0	0	0	0	0	
COUPLE	8, 7	P2=	.0500	1	2	3	4	5	6	7	8	9	10	AFFEC. REA.
COUPLE	8, 9	P2=	.0812	1	2	3	4	5	6	7	8	9	10	AFFEC. REA.
COUPLE	9, 1	P2=	.1625	0	0	0	0	0	0	0	0	0	0	
COUPLE	9, 2	P2=	.1625	0	0	0	0	0	0	0	0	0	0	
COUPLE	9, 3	P2=	.1562	0	0	0	0	0	0	0	0	0	0	
COUPLE	9, 4	P2=	.3937	0	0	0	0	0	0	0	0	0	0	
COUPLE	9, 5	P2=	.4187	0	0	0	0	0	0	0	0	0	0	
COUPLE	9, 6	P2=	.4375	0	0	0	0	0	0	0	0	0	0	
COUPLE	9, 7	P2=	.0562	1	2	3	4	5	6	7	8	9	10	AFFEC. REA.
COUPLE	9, 8	P2=	.0812	1	2	3	4	5	6	7	8	9	10	AFFEC. REA.

Table 6 ; (end)

	1	2	3	4	5	6	7	8	9
1	.0000	.0312	.0562	*.3250*	*.3625*	*.3812*	*.1250*	*.1375*	*.1312*
2	.0375	.0000	.0437	*.3187*	*.3562*	*.3500*	*.1250*	*.1375*	*.1312*
3	.0562	.0437	.0000	*.3187*	*.3562*	*.3500*	*.1250*	*.1375*	*.1312*
4	*.1375*	*.1125*	*.1250*	.0000	.0562	.0625	*.1437*	*.1625*	*.1375*
5	*.1375*	*.1250*	*.1250*	.0437	.0000	.0625	*.1437*	*.1625*	*.1375*
6	*.1312*	*.1250*	*.1250*	.0625	.0812	.0000	*.1437*	*.1500*	*.1187*
7	*.1562*	*.1500*	*.1500*	*.3937*	*.4187*	*.4375*	.0000	.0500	.0437
8	*.1562*	*.1500*	*.1562*	*.3937*	*.4187*	*.4375*	.0500	.0000	.0812
9	*.1625*	*.1625*	*.1562*	*.3937*	*.4187*	*.4375*	.0562	.0812	.0000

Table 7

- e_2 est en caractères droits, ex : .0375

- e_3 est en caractères italiques, ex : *.3250*

§3. AUTRES TYPES D'OPERATEURS LINGUISTIQUES

3.1. Nécessité de s'affranchir de certaines contraintes

L'un des inconvénients majeurs de l'opérateur qui a été décrit précédemment tient au fait que les formes à comparer doivent avoir exactement le même nombre de primitives du niveau considéré.

D'une façon générale, nous admettons qu'il existe une "mesure de dissemblance" entre la primitive A_i et la primitive A_j, $\varphi(A_j \mid A_i) \in \mathbb{R}$, telle que $\varphi(A_i \mid A_i) = 0$; mais $\varphi(A_j \mid A_i)$ n'est pas forcément égale à $\varphi(A_i \mid A_j)$ et de plus, l'hypothèse que φ est une distance n'est pas faite.

Ce type de mesure deux à deux est utilisé par les opérateurs de regroupement, comme nous l'avons vu au chapitre III.

Cette mesure de similarité permet d'ordonner la suite des noms A_j à partir du nom A_i, suivant les valeurs de $\varphi(A_j \mid A_i)$ donc d'obtenir un mot construit sur l'alphabet $\{\ldots A_i \ldots A_j \ldots\}$. L'opérateur va consister à obtenir comme résultats les n mots à partir des n points. Ces résultats seront bien entendu différents suivant le φ choisi. Nous serons guidés par l'invariance du mot en fonction des transformations géométriques.

Or dans les cas les plus courants, et en particulier pour l'application aux cartes RADAR, les éléments similaires ne sont pas forcément décrits par un même nombre d'éléments caractéristiques.

Si l'on se place au niveau des crêtes comme primitives on pourra avoir par exemple à comparer les 2 ensembles F1 et F2 ci-après (fig. IV.6).

$$F1 \rightarrow A_1\ B_1\ C_1\ D_1\ E_1\ F_1\ G_1$$

$$F_2 \rightarrow A_2\ B_2\ C_2\ D_2\ E_2\ F_2\ G_2$$

qui ne sont pas superposables mais dans lesquels il y a visiblement des sous-figures similaires. Par exemple,

$$D_1\ E_1\ F_1\ G_1 \quad \text{et} \quad C_2\ D_2\ F_2\ G_2$$

Nous avons donc cherché des opérateurs capables de comparer une forme F1 constituée de n1 primitives à une forme F2 possédant n2 primitives avec par exemple n1 < n2 ; sans avoir à examiner toutes les combinaisons C_{n2}^{n1} lorsqu'on ne recherche que l'inclusion éventuelle de F1 dans F2, et qui sont encore plus nombreuses si on veut trouver toutes les sous-figures de F1 qu'on peut retrouver dans F2.

Figure IV.6

Des programmes de ce genre avaient été mis au point par M. CAMILLERAPP [11] pour des ensembles de points superposables. Notre opérateur ne recherchera pas des identités mais des similarités en évitant la combinatoire.

3.2. Plan général d'étude des différents opérateurs utilisés

Dans ce qui suit nous allons décrire deux opérateurs construits à partir de principes très voisins mais ayant des propriétés différentes. Nous suivrons dans chaque cas le plan d'étude ci-après.

- définition de l'opérateur,
- propriétés par rapport aux transformations géométriques,
- possibilité de reconstituer une forme donnée,
- application à la reconnaissance de sous-figures ou "partielles" d'une figure donnée.

Chaque fois, nous montrerons expérimentalement qu'il existe un seuil au delà duquel une décision de similarité ne peut pas être prise avec certitude.

§4. UN OPERATEUR INVARIANT POUR LES ISOMETRIES

4.1. Définition de l'opérateur

Soit $X = \{A_1\ A_2 \ldots A_i \ldots A_n\}$ une forme constituée de n primitives représentées par des points de $\mathbb{R}^2$. On note d_{ij} la distance euclidienne des points A_i et A_j. L'opérateur $\&(X)$ est un ensemble de n mots de n lettres construits sur l'alphabet $\{A_i\}$ des noms des primitives de la manière suivante :

4.2. Propriétés de $\&(X)$

- Indépendance par rapport à la translation, la rotation, les déformations.
- Les renseignements que peut fournir l'ensemble $\{u_i\}$ sur les regroupements :

Si on considère le premier et le dernier (A_i et $A_{j_{n-1}}$) de la ligne i, on peut dire que tous les autres points de la figure donnée sont situés à l'intérieur de deux cercles :

le cercle de centre A_i de rayon $d_{ij_{n-1}}$

le cercle de centre $A_{j_{n-1}}$ de même rayon.

En effet, le mot $u_i = A_i\ A_{j_1} \ldots A_{j_{n-1}}$ correspond aux relations d'ordre :

$$A_i\ A_{j_k} \prec A_i\ A_{j_{n-1}} \qquad k \in [j_1, j_{n-2}]$$

En considérant simultanément les lignes i et j_{n-1} on peut répartir l'ensemble $\{A_i\}$ en 6 classes correspondant à 6 zones du plan représentées sur la figure IV.9.

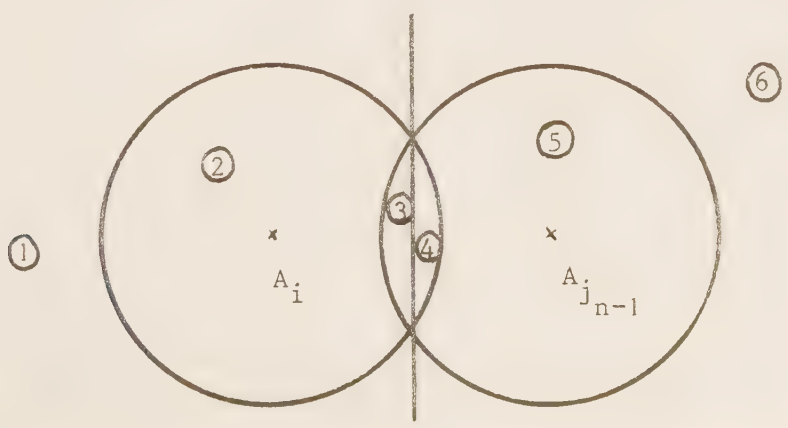


Figure IV.9

Tous les points de la ligne A_i autres que $A_{j_{n-1}}$ sont situés à l'intérieur du cercle de centre A_i de rayon $d_{ij_{n-1}}$:

Tous les points de la ligne $A_{j_{n-1}}$ situés entre $A_{j_{n-1}}$ et A_i sont à l'intérieur du cercle de centre $A_{j_{n-1}}$.

Ensuite, on répartit chaque point A_k à droite ou a gauche de la médiatrice de A_i $A_{j_{n-1}}$ suivant que dans la ligne A_k on rencontre $A_{j_{n-1}}$ avant A_i ou vice versa.

Ceci est vrai quel que soit la ligne i choisie. En recommençant la partition de $\{A_i\}$ en partant d'une autre ligne k, on obtient une nouvelle série de 6 zones du plan qui se combinent aux 6 premières. La localisation des groupes de points devient de plus en plus fine. Et très vite on peut avoir une idée des zones dense et des points isolés de la figure.

Un mot u_i a pour initiale la lettre A_i. Les (n - 1) autres lettres sont rangées après A_i dans un ordre tel que si $d_{ij} < d_{ik}$, A_j se trouve avant A_k. Si on a par exemple $d_{ij_1} < d_{ij_2} < d_{ij_3} \ldots < d_{ij_{n-1}}$.

Alors $$u_i = A_i\ A_{j_1}\ A_{j_2} \ldots A_{j_{n-1}}$$

Il y aura autant de mots u_i que de lettres dans $\{A_i\}$ et H(X) se présente donc sous la forme d'une matrice n x n.

Figure IV.7

1	5	2	6	4	3
2	1	5	6	3	4
3	4	6	5	2	1
4	6	5	3	1	2
5	6	4	1	2	3
6	5	4	1	3	2

Figure IV.8

Par exemple, la figure IV.7 est formée de 6 primitives dont les noms sont 1, 2, 3, ..., 6. La figure IV.8 fournit l'ensemble $\{u_i\}$ qui représente l'opérateur &(X) correspondant.

4.3. Algorithme de "découpage" du plan

Soit [MA(I, J)] la matrice correspondant à &(X) pour la figure IV.10 de 6 points.

Figure IV.10

1	4	5	6	2	3
2	3	**5**	6	4	1
3	2	5	6	4	1
4	1	**5**	6	2	3
5	6	4	2	3	1
6	5	4	3	2	1

matrice MA(I, J)

Au départ, prenons une ligne au hasard, par exemple la première. Elle définit les six zones décrites précédemment. Chacune des zones est repérée par un identificateur que nous appelons IDZON(I) et qui se présente sous la forme d'un mot de 32 bits :

A la première itération les 6 points vont être répartis dans les six zones (voir figure IV.9) ce qui se traduit par un tableau IDZON tel que :

- le bit 0 est à 1 si la zone est incluse dans le cercle de centre 3 et de rayon d_{13}.
- le bit i (avec $i \geqslant 1$) est à 1 si la zone est incluse dans le demi-plan défini par la médiatrice du couple A_k A_1 de points choisi pour la $i^{ème}$ itération, et le point A_k.

Pour l'exemple choisi, le tableau suivant est obtenu après 4 itérations.

IDZON(1) = 0 1 0 0 0 0 ...
IDZON(1) = 1 1 1 0 0 0 ...
IDZON(3) = 1 0 1 1 1 0 ...
IDZON(4) = 1 0 0 1 1 0 ...
IDZON(5) = 1 1 0 0 0 0 ...
IDZON(6) = 1 1 0 0 1 0 ...

La ligne 5 signifie par exemple que la plus petite zone contenant le sommet (5) est située à la fois :

- dans le cercle de centre (3) de rayon d_{31}.
- dans le demi-plan défini par la médiatrice du segment 13 et le point (1).
- dans le demi-plan défini par la médiatrice de 34 du côté du point (4).
- dans le demi-plan défini par la médiatrice de 46 et du côté de (6).

Remarques :

1. Au cours des itérations successives il est visible qu'un grand nombre de zones sont vides et que d'autres sont très denses. Le choix du couple pour l'itération suivante est alors fait dans la zone la plus dense si nous cherchons à effectuer la séparation la plus complète en un minimum d'itérations.

   On peut rapprocher ces considérations des procédés d'arborescence.

2. Le processus est considéré comme achevé lorsque toutes les lignes du tableau IDZON sont différentes, c'est-à-dire lorsqu'on a défini autant de zones non vides qu'il y a de points dans la figure donnée.

3.4. Reconstitution d'une forme à partir de sa représentation U_X

Supposons que l'on dispose de la matrice MA(I, J) qui représente &(X) et qu'à partir de cette matrice on ait construit le tableau IDZON et la liste des couples choisis successivement pour établir IDZON.

Soit $C_m(i, j)$, le couple utilisé à l'itération m.

A la première itération, si $C_1(1, 2)$ a été choisi, on peut définir 3 zones utiles dont les identificateurs sont

(1) 1 000 ...
(2) 1 100 ...
(3) 0 100 ...

(voir figure IV.11).

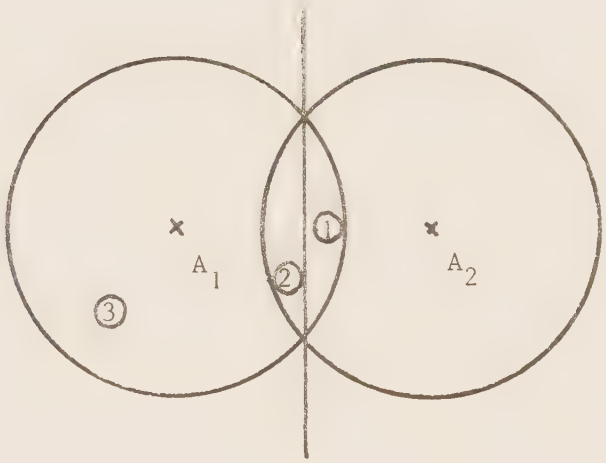


Figure IV.11.

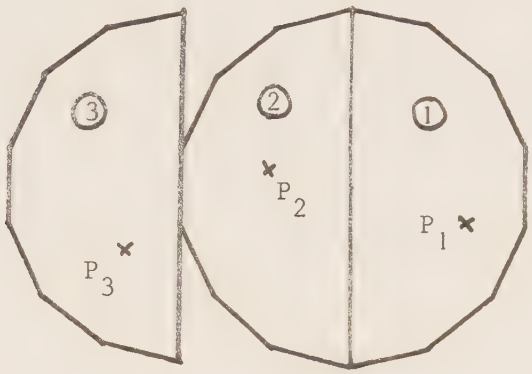
Les trois zones (1) (2) et (3) sont les seules, sur les 6 définies au total par la 1ère itération, qui ne soient pas vides.

A chacune de ces zones on attribue un point caractéristique, par exemple le barycentre. On a donc au départ, trois points P1, P2, P3 qui représentent les zones (1), (2), (3).

Algorithme de reconstitution

Pour la commodité des calculs, on a ramené chaque zone à un polygone et on la représente en mémoire par les coordonnées de ses sommets.

Au départ, les zones (1) , (2) et (3) ont été simplifiées de la manière suivante :



et les points P1, P2 et P3 sont pris au hasard dans chaque zone afin d'amorcer un processus de séparation convergent. En effet, si on prenait les **barycentres** de ces zones, les médiatrices définies ultérieurement seraient toutes parallèles.

Remarque : la méthode de reconstitution impose des propriétés à la figure reconstituée ; dans le cas envisagé les points successifs sont les centres de gravité de certaines zones ; dans la reconstitution à partir du résultat du prochain opérateur **envisagé**, les points de l'enveloppe convexe sont uniformément répartis sur un cercle. Il faudrait donc qu'il existât toujours une transformation, laissant u_X invariant, qui permette de passer de la figure origine à la figure reconstituée. Il est bien clair que, dès que les figures ont un grand nombre de points, une telle transformation ne peut être trouvée. C'est la raison pour laquelle la reconstitution est limitée à un certain nombre de points.

On met en place les trois points P1, P2, P3 et aussi le premier couple C(1, 2) sur deux des trois points précédents.

On trace la médiatrice de C(1, 2) et on détermine l'intersection de cette médiatrice avec les zones déjà en place.

D'une manière générale, à l'étape i, ou détermine les intersections de la médiatrice de $C_i(k, 1)$ avec les zones de l'étape i - 1.

On place alors le point (1) au point caractéristique de la zone qui lui correspond parmi les nouvelles zones créées.

On itère le processus jusqu'à ce que la liste $C_i(i, j)$ soit épuisée. On aura alors rencontré toutes les zones contenant un point de la figure donnée.

Par construction, la figure Y reconstituée est telle que $\mathcal{E}(Y) = \mathcal{E}(X)$.

La figure IV.12 montre le résultat de la reconstitution de la figure IV.10.

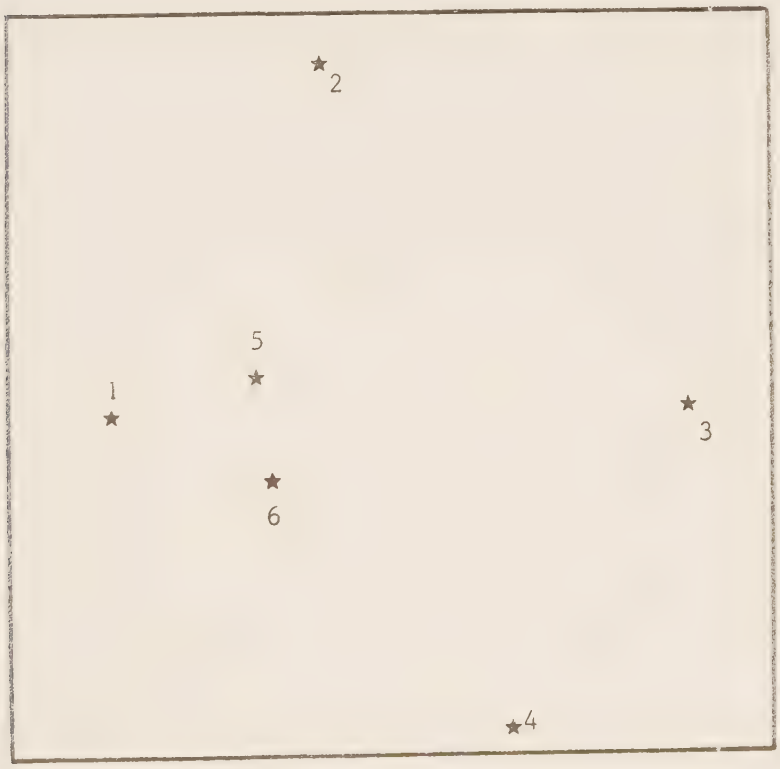


Figure IV.12

4.4. Méthode de comparaison de deux formes

Considérons deux figures, l'une A, de NA points, représentée par la matrice MA, l'autre B, de NB points, représentée par la matrice MB (on supposera NA $\leqslant$ NB).

Notre problème est de déterminer, uniquement à l'aide des matrices MA et MB, si une sous figure de A peut se retrouver dans la figure B.

Figure A

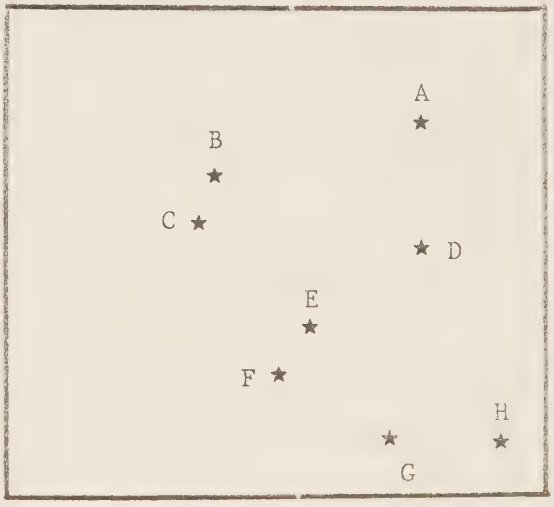


Figure B

A	D	E	F	B	C	
B	C	E	F	D	A	
C	B	E	F	D	A	
D	A	E	F	B	C	
E	F	D	B	C	A	
F	E	D	C	B	A	

matrice MA

A	D	E	F	B	C	H	G
B	C	E	F	D	A	G	H
C	B	E	F	D	G	A	H
D	A	E	F	B	C	H	G
E	F	D	G	H	B	C	A
F	E	G	H	D	C	B	A
G	F	H	E	C	D	B	A
H	G	F	E	D	A	C	B

matrice MB

Le point B de MA peut correspondre aux joints B, C, D de MB et à eux seuls.

D'une façon générale, pour que MB(k, 1) corresponde à MA(i, 1) on doit avoir :

$$\sup(1, NAB - NA + i) \leqslant k \leqslant \inf(NB, NB - NAB + i)$$

où NAB est le nombre de points de la figure partielle possible. Mais MA(i, 1) existe dans chacune des lignes de MA. Soient μ et ν ses coordonnées.

Chaque point MA(μ, ν) correspond à un point MB(t, s) dans MB, tel que, $\forall t$:

$$\text{si } \mu < i \quad \sup(1, NAB - NA + \mu) \leqslant t \leqslant \inf(k - 1, NB - NAB + \mu)$$

$$\text{si } \mu > i \quad \sup(k + 1, NAB - NA + \mu) \leqslant t \leqslant \inf(NB, NB - NAB + \nu)$$

et d'autre part,

$$\sup(2, NAB - NA + \nu) \leqslant s \leqslant \inf(NB, NB - NAB + \nu)$$

C'est-à-dire que chaque MB(t, s) se trouve dans un rectangle défini par ces inégalités et au total il y a(NA - 1)rectangles de ce type dans MB.

Toutes les affectations possibles des points MA sur les points de MB pour des partiels de NAB points peuvent être matérialisées par un 1 dans un tableau de la forme :

Exemple avec Na = 6, Nb = 8.

MA \ MB	A	B	C	D	E	F	G	H
A	1	0	0	0	0	0	0	0
B	0	1	0	0	0	0	0	0
C	0	0	1	0	0	0	0	0
D	0	0	0	1	0	0	0	0
E	0	0	0	0	1	0	0	0
F	0	0	0	0	0	1	1	1

On voit ici qu'il y a trois solutions possibles :

A-A	A-A	A-A
B-B	B-B	B-B
C-C	C-C	C-C
D-D	D-D	D-D
E-E	E-E	E-E
F-F	F-G	F-H

Le dénombrement de solution à partir de ce tableau est expliqué plus loin.

Il reste maintenant à vérifier ces solutions : soit par l'application inverse Γ^{-1}, soit en comparant les sous matrices des distances qui en résultent. Nous verrons ces deux méthodes.

Dénombrement des solutions

Exemple : NA = 5, NB = 6.

COM =

		MB1	MB2	MB3	MB4	MB5	MB6
		A'	D'	E'	B'	C'	F'
MA1	C	1	1	1	0	0	0
MA2	D	0	1	1	0	0	0
MA3	B	0	0	1	1	0	0
MA4	A	0	0	0	1	1	0
MA5	E	0	0	0	0	1	1

De toutes les affectations possibles pour chaque point, il faut trouver les affectations possibles de NAB points de MA sur NAB points de MB.

L'ordre des points devant être conservé, cela revient à chercher les suites croissantes de NAB couples (α, β) telles que la case $(MA\alpha)$ $(MA\beta)$ contienne un "1".

Exemple : C D B A E
A' D' E' B' C'

Il y en a d'autres ; nous nous proposons d'en faire la liste exhaustive. Indiquons l'algorithme avant de l'appliquer à l'exemple :

1. Détermination de COM réduit aus points choisis pour le partiel de NAB points dans MA.
2. Détermination de la 1ere solution (suite croissante de NAB termes) jusqu'au NAB^{ième} point de MA. Soit j la colonne correspondant au point de MB qui lui est affecté. On pose k = 1.
3. Si $j \leqslant NB$ tous les 1 de NAB une ligne de COM entre j et NB donne une solution.

4. Si j = NB + 1 on va à la ligne NAB - k, on cherche la colonne jj contenant le -1 de la solution précédente et on prend le '1' suivant compris entre jj + 1 et NB. On réforme alors une suite croissante de NAB termes ce qui donne une solution.
5. S'il n'est plus possible d'atteindre une suite de NAB termes, l'algorithme est terminé. Sinon on fait k = k + 1 et on retourne à l'action 3.

Appliquons ceci à notre exemple avec NAB = 4. On suppose avoir choisi le partiel de MA formé par les 4 premiers points.

1) COM réduit s'écrit :

	A'	D'	E'	B'	C'	F'
C	1	1	1	0	0	0
D	0	1	1	0	0	0
B	0	0	1	1	0	0
A	0	0	0	1	1	0

2) D'où la première solution :

	A'	D'	E'	B'	C'	F'
C	1					
D		1				
B			1			
A				1		

Ce qui correspond à

C → A'

D → D'

B → E'

A → B'

3) Autres solutions :

	A'	D'	E'	B'	C'	F'
C	1					
D		1				
B			1			
A					1	

C → A'

D → D'

B → E'

A → C'

4)

	A'	D'	E'	B'	C'	F'
C	1					
D		1				
B				1		
A					1	

C → A'

D → D'

B → B'

A → C'

Toujours dans la phase 4 on a :

•

	A'	D'	E'	B'	C'	F'
C	1					
D			1			
B				1		
A					1	

C → A'

D → E'

B → B'

A → C'

	A'	D'	E'	B'	C'	F'
C		1				
D			1			
B				1		
A					1	

C → D'

D → E'

B → B'

A → C'

Il n'y a plus de possibilités de former des suites croissantes de NAB termes : il y a 5 solutions :

Exemple avec NA = 6, NB = 8, NAB = 5.

ICOMP

	7	1	2	5	8	3	6	4
6	1	1	1	1	0	0	0	0
3	1	1	1	1	1	0	0	0
2	0	1	1	1	1	1	0	0
5	0	0	1	1	1	1	1	0
4	0	0	0	1	1	1	1	1
1	0	0	0	0	1	1	1	1

SOLUTION NO = 1

	7	1	2	5	8	3	6	4
6	1	0	0	0	0	0	0	0
3	0	1	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0
5	0	0	0	1	0	0	0	0
4	0	0	0	0	1	0	0	0
1	0	0	0	0	0	0	0	0

SOLUTION NO = 2

	7	1	2	5	8	3	6	4
6	1	0	0	0	0	0	0	0
3	0	1	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0
5	0	0	0	1	0	0	0	0
4	0	0	0	0	0	1	0	0
1	0	0	0	0	0	0	0	0

Résultats avec l'opérateur Γ

Il y a C_{NA}^{NAB} façons de choisir des partiels de NAB points dans la figure MA et pour chacun de ces partiels il y a C_{NB}^{NAB} façons d'essayer de les affecter dans MB. Au total, le maximum de solutions possibles (combinatoire) serait $C_{NA}^{NAB} \times C_{NB}^{NAB}$.

Au lieu de faire l'étude sur tous les partiels de MA, nous l'avons faite sur un partiel au hasard après avoir vérifié que les résultats n'étaient pas modifiés au facteur C_{NA}^{NAB} près (voir annexe 1).

NA = 10
NB = 12

NAB	6	7	8	9	10
nombre d'affect.	924	708	168	0	0
Combinatoire	924	792	495	220	66

NA = 9
NB = 11

NAB	4	5	6	7	8	9
nombre d'affect.	330	462	462	155	0	0
combinatoire	330	462	462	330	165	55

La comparaison de sous-figures en n'utilisant que l'application Γ (sans faire de vérifications) nous a donné un très grand nombre de solutions qui, bien que s'écartant sensiblement de la combinatoire pour NAB voisin de NA, reste encore trop élevé. Nous avons restreint l'ensemble des solutions déterminées par Γ en utilisant la vérification par l'application inverse Γ^{-1}.

Vérification par Γ^{-1}

Cette vérification se fait point par point.

Figure

L'application Γ^{-1} définit dans MA (NB - 1)rectangles et il faut trouver le point affecté dans au moins NAB - 1 de ces rectangles.

Détermination des bornes des rectangles.

Soient (μ, ν) les coordonnées des points tels que MB(k, 1) = MB(μ, ν) (on a supposé que l'affectation MA(i, 1) → MB(k, 1) a été acceptée par Γ)

Les rectangles sont déterminés par s et t. Les conditions sur s sont :

$$\sup(e, na - nb + \nu) \leqslant s \leqslant \inf(na, \nu)$$

et celles sur t :

$\mu < k$ $\quad \sup(1, na - nb + \mu) \leqslant t \leqslant \inf(i - 1, \mu)$

$\mu > k$ $\quad \sup(i + 1, na - nb + \mu) \leqslant t \leqslant \inf(na, \mu)$

Le dénombrement des solutions de vérification par Γ^{-1} donne un nombre beaucoup plus restreint de solutions qu'en utilisant Γ seul (voir exemples).

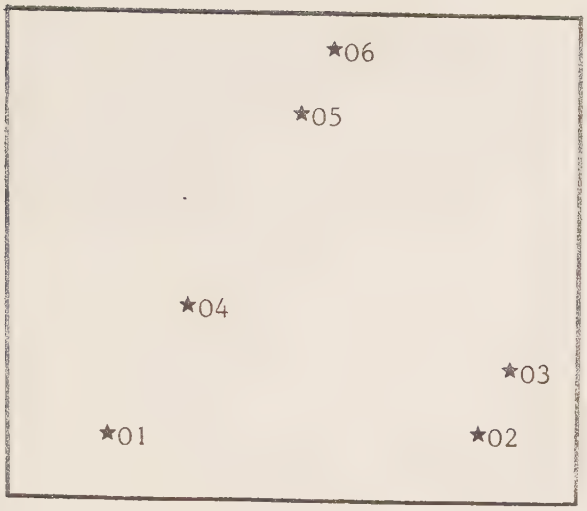
Nous avons également essayé de vérifier les solutions de la manière suivante :

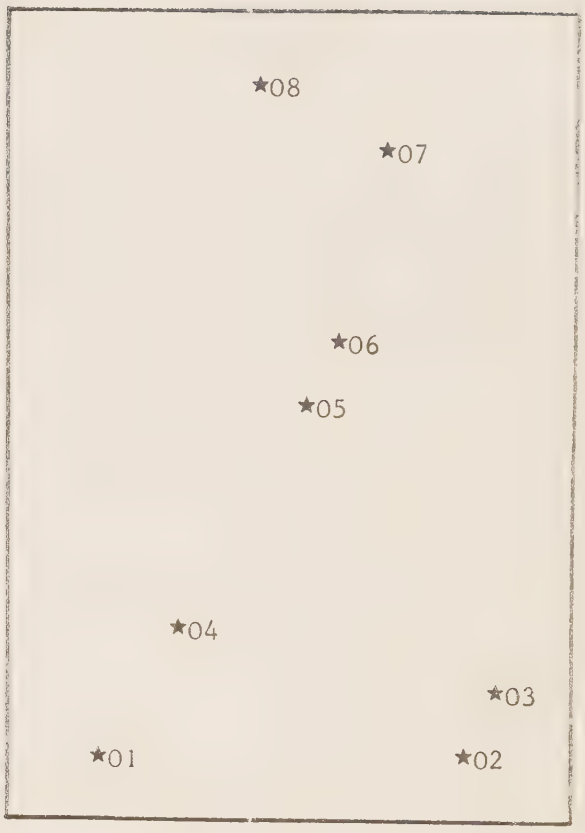
A partir d'une affectation déterminée par Γ, nous construisons les sous-matrices de NAB points tirées des matrices MA et MB. Leur comparaison en liaison avec le tableau des affectations permet une vérification sur l'ensemble de l'affectation.

Cette vérification apparaît plus stricte que Γ^{-1} qui ne vérifie que point par point. Les résultats expérimentaux ont vérifié cette hypothèse.

4.6. Résultats expérimentaux

NA = 6, NB = 8 (avec une figure égale).





MATRICE MA

1	4	5	6	2	3
2	3	5	6	4	1
3	2	5	6	4	1
4	1	5	6	2	3
5	6	4	3	3	1
6	5	4	2	2	1

MATRICE MB

1	4	5	6	2	3	8	7
2	3	5	6	4	1	7	8
3	2	5	6	4	7	1	8
4	1	5	6	2	3	8	7
5	6	4	7	8	2	3	1
6	5	7	8	4	3	2	1
7	6	8	5	3	4	2	1
8	7	6	5	4	1	3	2

Vérification par Γ^{-1}

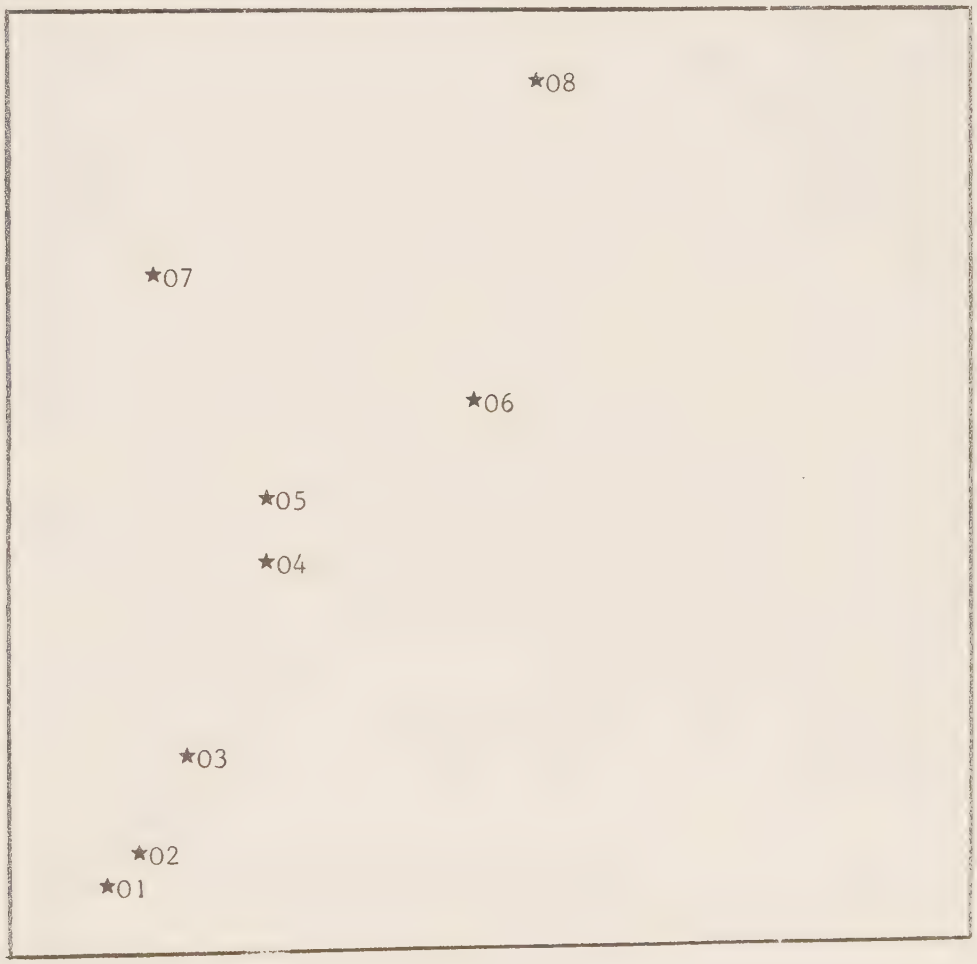
NAB	1	2	3	4	5	6
nombre de sol.	8	28	32	37	2	1
nombre de sol. éliminées par Γ^{-1}	0	0	24	33	5	2

On retrouve bien pour NAB = 6 une solution.

Vérification par les sous-matrices

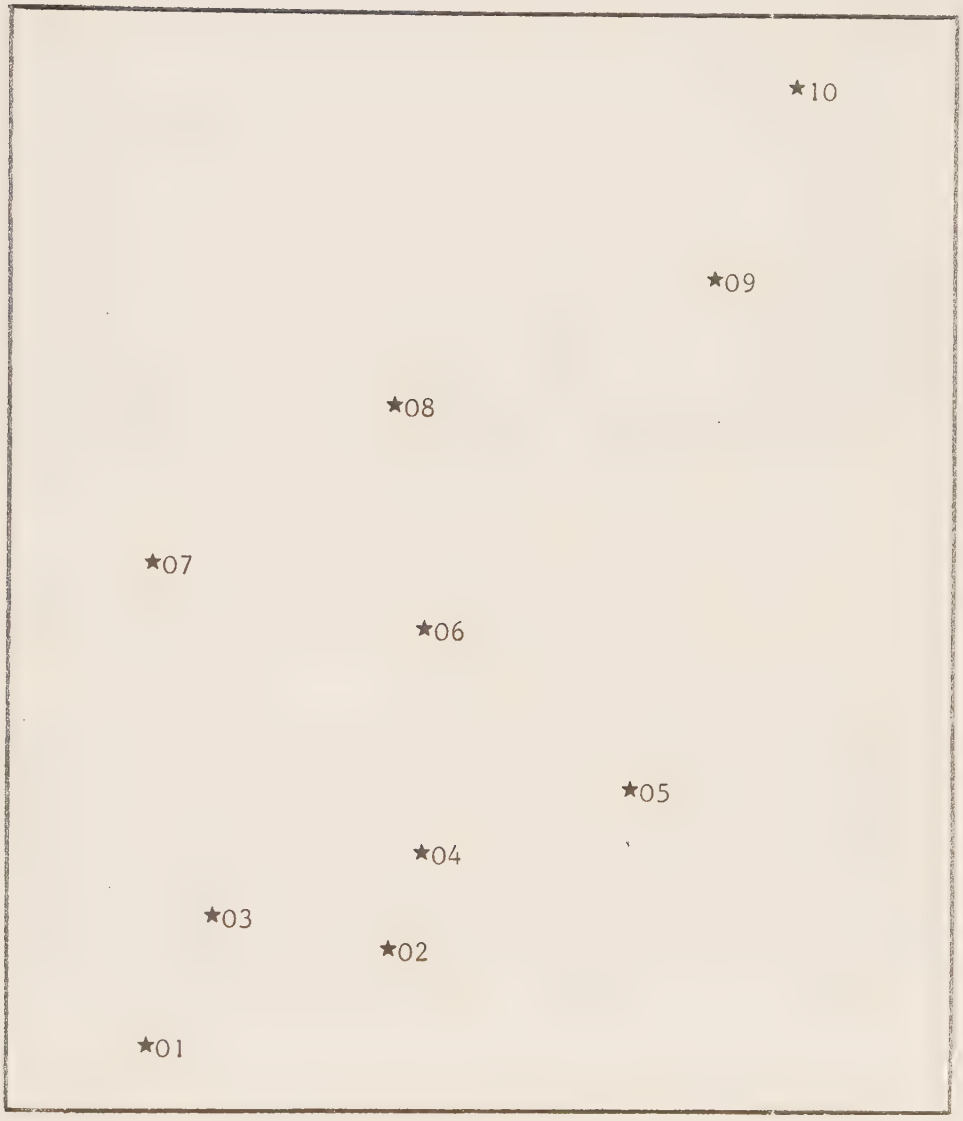
```
  1 AFFECTATIONS POSSIBLES POUR NAB=  6  IL Y EN A EU     2 ELIMINEES PAR G**-1
  1 AFFECTATIONS POSSIBLES POUR NAB=  5  IL Y EN A EU     6 ELIMINEES PAR G**-1
  1 AFFECTATIONS POSSIBLES POUR NAB=  4  IL Y EN A EU    69 ELIMINEES PAR G**-1
 23 AFFECTATIONS POSSIBLES POUR NAB=  3  IL Y EN A EU    33 ELIMINEES PAR G**-1
 28 AFFECTATIONS POSSIBLES POUR NAB=  2  IL Y EN A EU     0 ELIMINEES PAR G**-1
  8 AFFECTATIONS POSSIBLES POUR NAB=  1  IL Y EN A EU     0 ELIMINEES PAR G**-1
```

NA = 8, NB = 10 et pas de sous figure égale à NAB = 8.



MATRICE MA

1	2	3	4	5	7	6	8
2	1	3	4	5	7	6	8
3	2	1	4	5	7	6	8
4	5	3	2	6	7	1	8
5	4	3	7	6	2	1	8
6	5	4	8	3	7	2	1
7	5	4	3	6	2	1	8
8	6	5	7	4	3	2	1



MATRICE MB

1	3	2	7	4	6	8	5	9	10
2	4	3	6	1	5	8	7	9	10
3	1	2	7	4	6	8	5	9	10
4	2	6	3	5	8	1	7	9	10
5	4	6	2	8	9	3	10	7	1
6	4	8	2	5	3	7	1	9	10
7	3	8	1	6	4	2	5	9	10
8	6	4	7	2	5	3	9	1	10
9	10	5	8	6	4	2	7	3	1
10	9	8	5	6	4	2	7	3	1

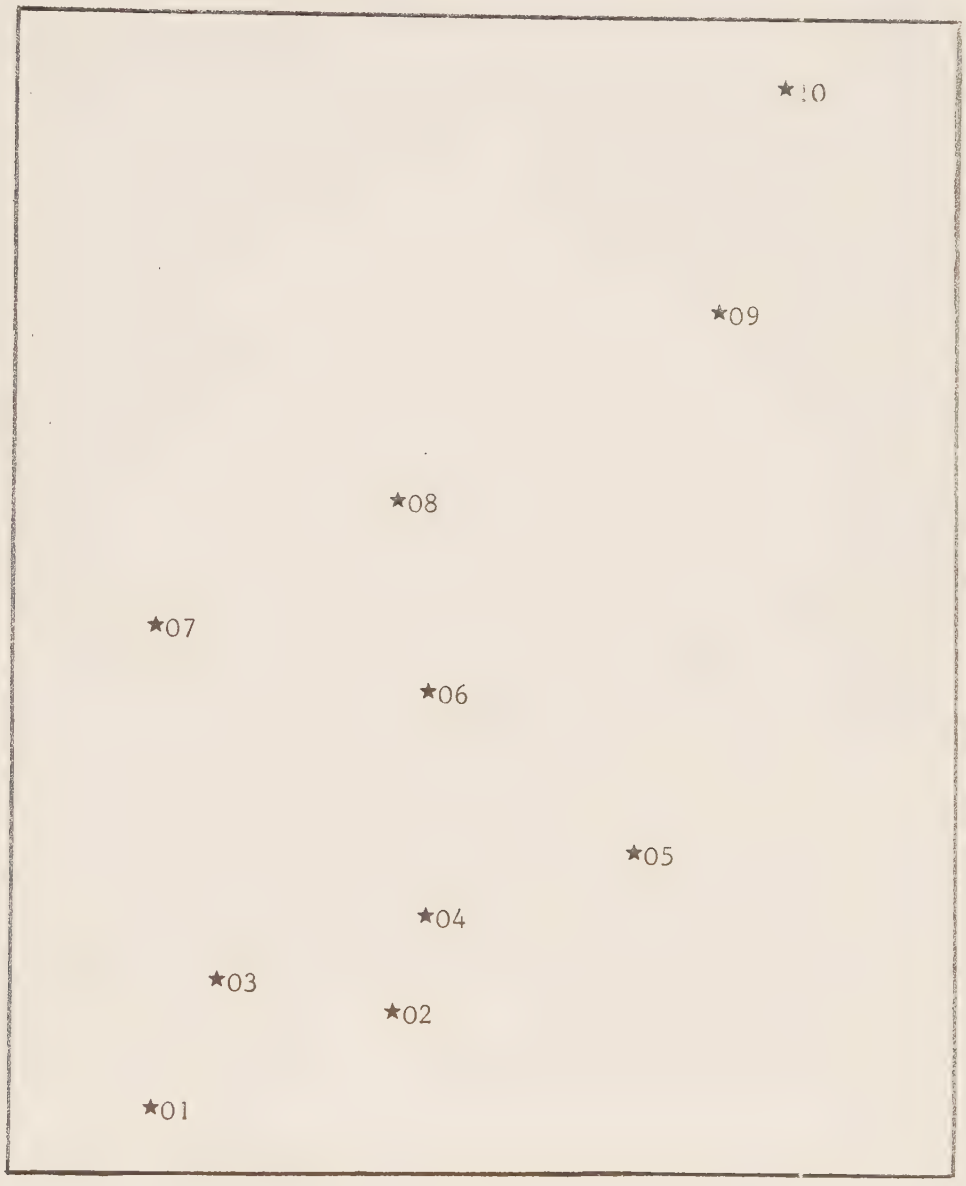
Vérification par Γ^{-1}

NAB	1	2	3	4	5	6	7	8
nombre de sol.	10	45	67	62	15	15	0	0
nombre de sol. éliminées par Γ^{-1}	0	0	53	148	237	195	36	0

Vérification par les sous matrices

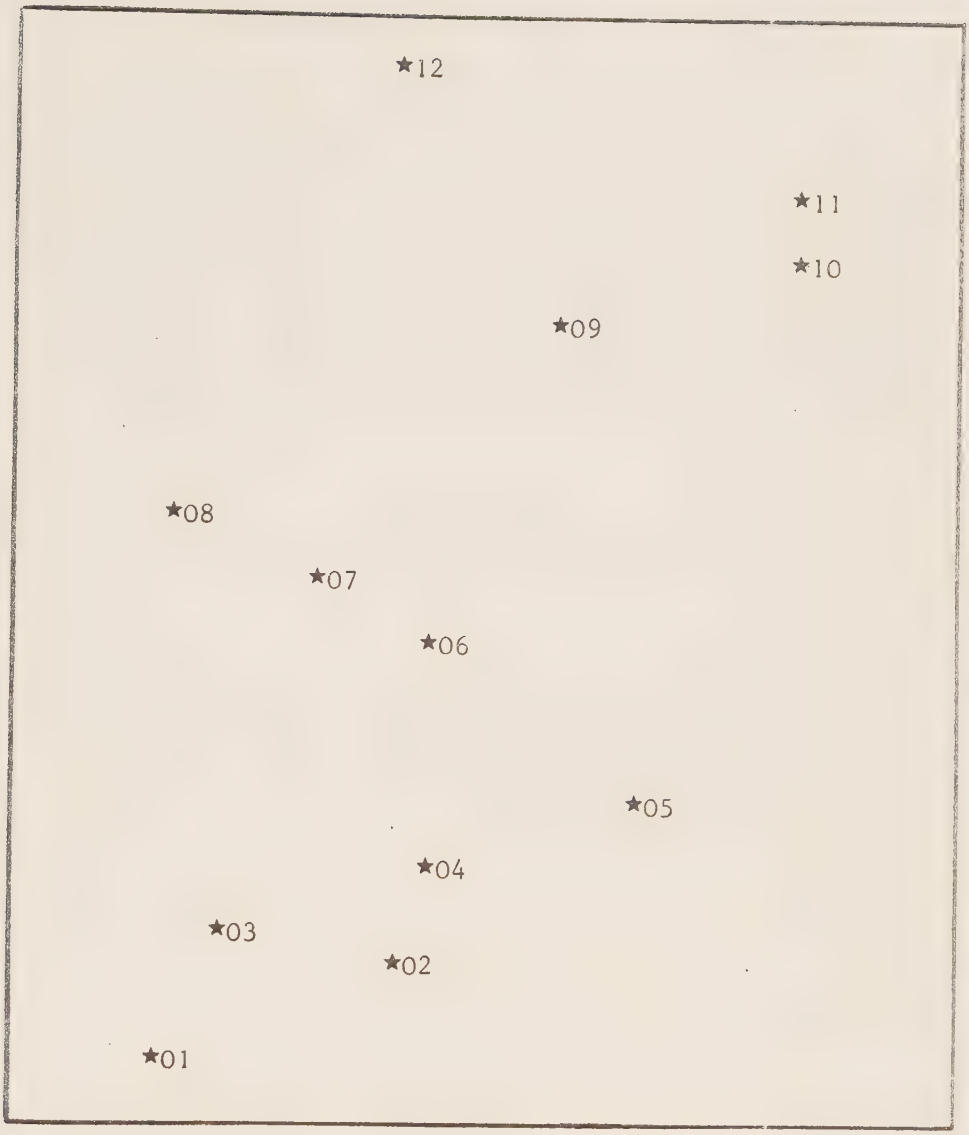
```
 0 AFFECTATIONS POSSIBLES POUR NAB=  8  IL Y EN A EU     0 ELIMINEES PAR G**-1
 0 AFFECTATIONS POSSIBLES POUR NAB=  7  IL Y EN A EU    36 ELIMINEES PAR G**-1
 0 AFFECTATIONS POSSIBLES POUR NAB=  6  IL Y EN A EU   210 ELIMINEES PAR G**-1
 9 AFFECTATIONS POSSIBLES POUR NAB=  5  IL Y EN A EU   243 ELIMINEES PAR G**-1
22 AFFECTATIONS POSSIBLES POUR NAB=  4  IL Y EN A EU   188 ELIMINEES PAR G**-1
58 AFFECTATIONS POSSIBLES POUR NAB=  3  IL Y EN A EU    62 ELIMINEES PAR G**-1
45 AFFECTATIONS POSSIBLES POUR NAB=  2  IL Y EN A EU     0 ELIMINEES PAR G**-1
10 AFFECTATIONS POSSIBLES POUR NAB=  1  IL Y EN A EU     0 ELIMINEES PAR G**-1
```

NA = 10, NB = 12 et nous sommes sûrs que pour NAB = 6, nous avons une solution.



MATRICE MA

1	3	2	7	4	6	8	5	9	10
2	4	3	6	1	5	8	7	9	10
3	1	2	7	4	6	8	5	9	10
4	2	6	3	5	8	1	7	9	10
5	4	6	2	8	9	3	10	7	1
6	4	8	2	5	3	7	1	9	10
7	3	8	1	6	4	2	5	9	10
8	6	4	7	2	5	3	9	1	10
9	10	5	8	6	4	2	7	3	1
10	9	8	5	6	4	2	7	3	1



MATRICE MB

1	3	2	4	8	7	6	5	9	12	10	11
2	4	3	6	7	1	5	8	9	12	10	11
3	1	2	4	7	8	6	5	9	12	10	11
4	2	6	7	3	5	1	8	9	12	10	11
5	4	6	2	9	7	10	11	3	8	12	1
6	4	7	2	5	9	3	8	12	1	10	11
7	6	8	4	3	2	9	1	12	5	10	11
8	7	3	6	1	4	2	2	9	5	10	11
9	12	6	10	11	7	5	4	8	2	3	1
10	11	9	5	12	6	4	7	2	8	3	1
11	10	9	5	12	6	7	4	2	8	3	1
12	9	7	8	6	11	10	4	5	2	3	1

Vérification par les sous matrices

```
 0 AFFECTATIONS POSSIBLES POUR NAB= 10   IL Y EN A EU      0 ELIMINEES PAR G**-1
 0 AFFECTATIONS POSSIBLES POUR NAB=  9   IL Y EN A EU     20 ELIMINEES PAR G**-1
 0 AFFECTATIONS POSSIBLES POUR NAB=  8   IL Y EN A EU    435 ELIMINEES PAR G**-1
 0 AFFECTATIONS POSSIBLES POUR NAB=  7   IL Y EN A EU    792 ELIMINEES PAR G**-1
 1 AFFECTATIONS POSSIBLES POUR NAB=  6   IL Y EN A EU    923 ELIMINEES PAR G**-1
 1 AFFECTATIONS POSSIBLES POUR NAB=  5   IL Y EN A EU    791 ELIMINEES PAR G**-1
 3 AFFECTATIONS POSSIBLES POUR NAB=  4   IL Y EN A EU    492 ELIMINEES PAR G**-1
12 AFFECTATIONS POSSIBLES POUR NAB=  3   IL Y EN A EU    208 ELIMINEES PAR G**-1
66 AFFECTATIONS POSSIBLES POUR NAB=  2   IL Y EN A EU      0 ELIMINEES PAR G**-1
12 AFFECTATIONS POSSIBLES POUR NAB=  1   IL Y EN A EU      0 ELIMINEES PAR G**-1
```

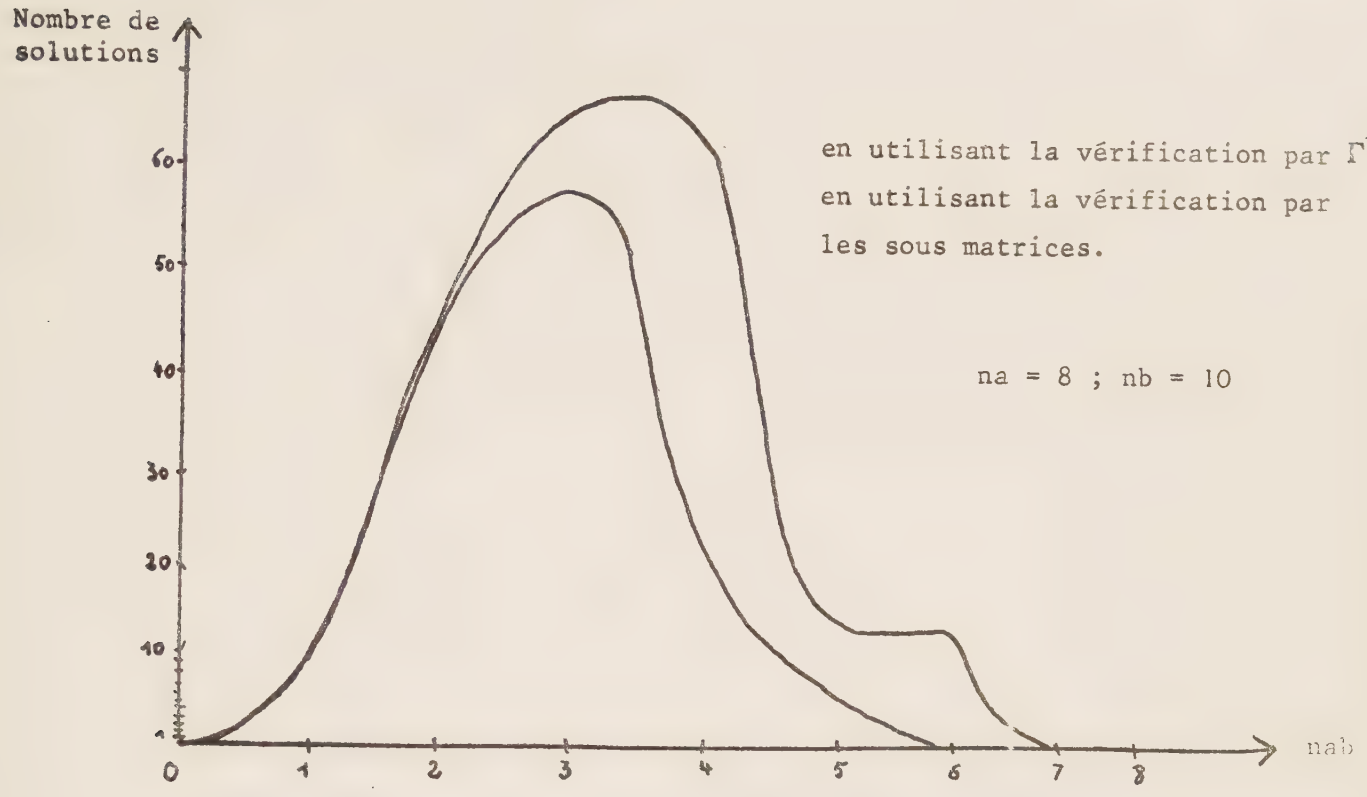
La solution que nous prévoyions pour NAB = 6 se retrouve bien après la vérification.

Remarque : la comparaison des figures avec la vérification par la reconstitution des sous-matrices présente un autre avantage sur la comparaison avec la vérification par Γ^{-1} : celui d'une plus grande rapidité d'exécution en machine.

Sur la 360 91, pour MA = 10, MB = 12 le temps d'exécution est de 40,29 s. pour le programme utilisant les sous matrices.

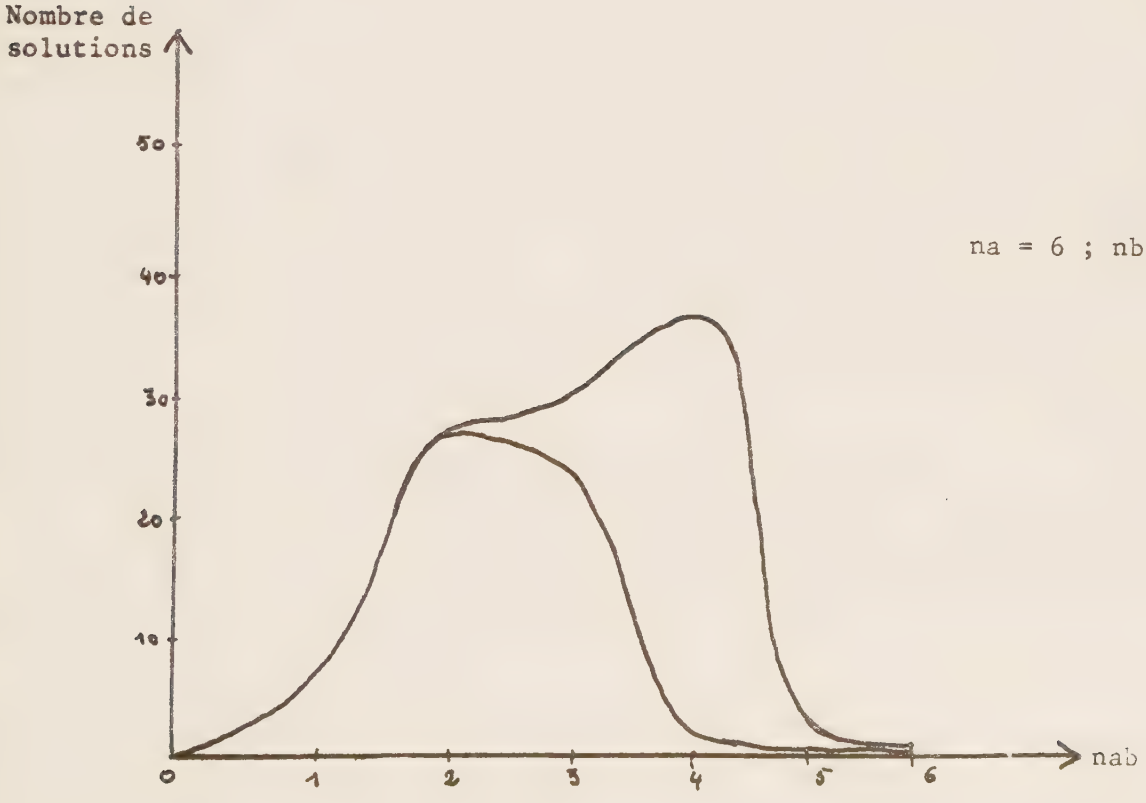
Pour MA = 6, MB = 8 et MA = 8, MB = 10 le temps d'exécution pour le même programme est de 7,35 s. tandis que pour le programme utilisant Γ^{-1} le temps est sensiblement plus long.

Ces résultats nous donnent les graphes suivants



en utilisant la vérification par Γ^{-1}
en utilisant la vérification par les sous matrices.

na = 8 ; nb = 10



na = 6 ; nb = 8

A N N E X E 1

Utilisation des programmes

RAPPEL : nous disposons de deux figures :

- l'une appelée "catalogue" de NB points,
- l'autre appelée "photo" de NA points (NA $\leqslant$ NB).

On choisit des partiels de NAB points dans la "photo" (NAB $\leqslant$ NA) et l'on recherche toutes affectations possibles de ces NAB points de photo sur les points du catalogue.

Les résultats obtenus sont :

- une image des 2 figures photo et catalogue,
- les matrices représentatives de ces figures. La matrice photo étant parenthésée.
- le nombre de partiels de NAB points existent dans le catalogue et la photo.
- la courbe représentant la variation du nombre de partiels de NAB points existent dans les deux figures lorsque NAB varie de 2 à NA.

Le programme comprend trois parties logiques :

1. Détermination de la matrice des distances pour la photo avec parenthésage. Détermination de la matrice des distances du catalogue sans parenthésage par la subroutine COORD.

2. Constitution du tableau de toutes les affectations possibles par l'algorithme de Renard. Dénombrement et distinction de chacune des solutions possibles.
3. Pour chacune des solutions possibles d'affectation pour des partiels de NAB points de la photo sur le catalogue, vérification par la comparaison des deux sous-matrices de NAB points reconstruites.

Nous allons voir comment sont introduites les données, puis les contraintes de format pour les sorties et enfin le rôle de chaque sous-programme.

§1. DESCRIPTION DES DONNEES

Le nombre de points de la matrice MA (photo) est lu dans le format I2 (1ère carte de donnée).

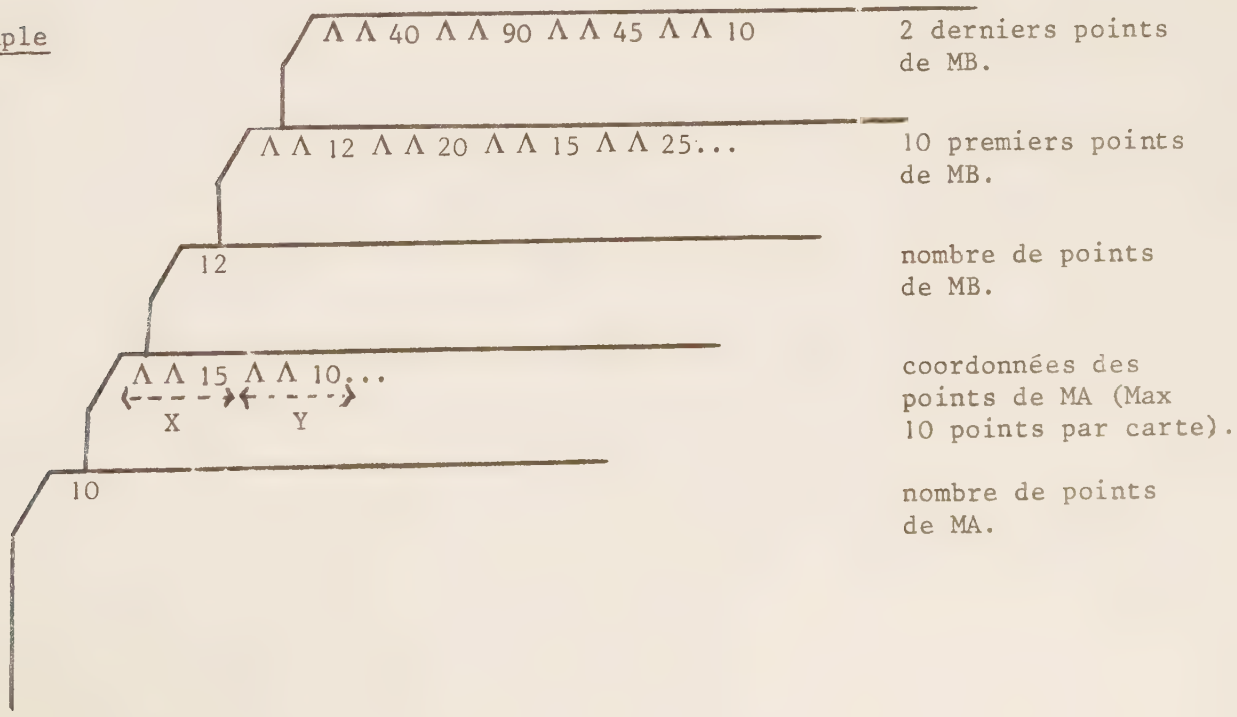
Puis viennent les coordonnées des points de MA, lues dans le format I4 : pour chaque point, abscisse puis ordonnée, ce qui fait 10 points par carte.

Le nombre de points de la matrice MB (catalogue) est lu dans le format I2.

Les coordonnées des points de MB sont alors lues en I4 comme précédemment.

Nota : les points doivent être entrés suivant les ordonnées croissantes.

Exemple



§2. CONTRAINTES POUR LES SORTIES

1. Sortie de la matrice MA parenthèsée : dépend du nombre de points de MA ; c'est le paramètre NA. Le format à modifier est le format 2002 : exemple : si NA = 6 on a

2002 FORMAT (1X, 12I3)

soit d'une façon générale 2 fois la valeur de NA.

2. Sortie de la matrice MB (catalogue) : dépend du nombre de points de MB : c'est le paramètre NB. Le format à modifier est le format 1005 : exemple : si NB = 8 on a

1005 FORMAT ('0', 8I3)

soit d'une façon générale la valeur de NB.

§3. FONCTIONS DE CHAQUE SOUS-PROGRAMME

3.1. Le sous-programme COORD

A partir des coordonnées des points, il calcule la matrice des distances sans parenthèsage (ou éventuellement avec un parenthésage par trait d'union indiquant que certains points sont à égale distance :

Exemple : 1 3 2 - 5 - 4 6 7 signifie que les points 2, 5, 4 sont à égale distance du point 1. Pour obtenir ce parenthésage il suffit d'enlever la boucle faisant appel à la fonction IABS de l'étiquette 502 à l'étiquette 70).

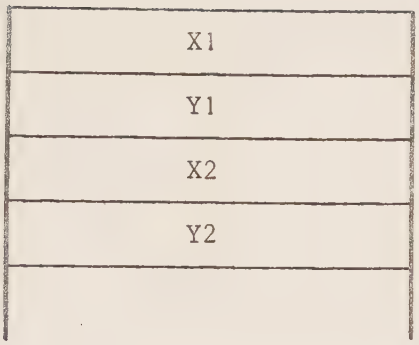
Les paramètres sont : COORD (MOT, N, PO).

MOT est la matrice des distances désirée (max 20, 20)

N est le nombre de points (max 20)

PO est le tableau contenant les coordonnées des points (PO (40))

PO



abscisse du 1er point

ordonnée du 1er point

abscisse du 2ème point

ordonnée du 2ème point

3.2. Le sous-programme TRACE permet de visualiser le contenu d'un tableau comme PO décrit précédemment.

TRACE(TAB, N)

TAB : nom du tableau où sont les coordonnées (Max TAB(60))

N : nombre de points à prendre dans ce tableau.

Les points sont numérotés dans l'ordre d'inscription dans le tableau TAB.

```
//JLB       JOB S*DESIN1205*GRV***0400P004.0*0000*010*250*91I*00*
//       EXEC FORTGCLG
//FORT.SYSIN DD *
      INTEGER MA1(20,20)
      INTEGER ECRMAT(60)
      INTEGER AVIRER(20),SMA(20,20)
      INTEGER SMB(20,20)
      INTEGER MOT(20,20),MB2(20,20),JEGAL(20,2),ANAB
      INTEGER PO2(40),PO3(40)
      INTEGER MB(20,20),ICOMP(20,20),ICOMK(20,20)
      INTEGER  GARD(2,20),SAUV(20),MA(20,40),PO(40),IDIST(20,20),EPS
      LR=5
      LW=6
      INDIC=0
  188 READ(LR,2000)NA
      NN=2*NA
 2000 FORMAT(I2)
      READ(LR,2001)(PO(I),I=1,NN)
 2001 FORMAT(20I4)
      DO 126 I=1,NA
      DO 126 J=1,NN
  126 MA(I,J)=0
      DO 102 I=1,NA
      LJ=I-1
      IF(LJ.EQ.0)GO TO 102
      DO 103 J=1,LJ
      II=2*I
      JJ=2*J
      MI=II-1
      MJ=JJ-1
      IDIST(I,J)=(PO(MI)-PO(MJ))*(PO(MI)-PO(MJ))+(PO(II)-PO(JJ))*(PO(II)
     1-PO(JJ))
      IDIST(J,I)=IDIST(I,J)
  103 CONTINUE
      IDIST(I,I)=1000000
  102 CONTINUE
      IDIST(1,1)=1000000
      EPS=0
      DO 104 I=1,NA
      DO 116 L=1,NA
      MIN=1000000
      DO 100 J=1,NA
      IF(IDIST(I,J).GE.MIN)GO TO 100
      GARD(1,L)=IDIST(I,J)
      GARD(2,L)=J
      MIN=IDIST(I,J)
      JJ=J
  100 CONTINUE
      IDIST(I,JJ)=1000000
  116 CONTINUE
      J=1
  113 DO 101 L=1,20
  101 SAUV(L)=0
      K=0
  210 L=J+1
```

```
      IF(L.GE.NA) GO TO 124
  110 IEC=GARD(1,L)-GARD(1,J)
      IF(IEC.GE.EPS)GO TO 106
      IF(L.EQ.(J+1))GO TO 107
      K=K+1
      SAUV(K)=GARD(2,L)
      GO TO 108
  107 K=K+1
      SAUV(K)=GARD(2,J)
      K=K+1
      SAUV(K)=GARD(2,L)
  108 L=L+1
      IF(L.NE.NA) GO TO 110
      IF(J.NE.1) GO TO 112
      MA(I,1)=I
      MA(I,2)=1
      IL=3
      KK=1
  123 IF(SAUV(KK).NE.0)GO TO 111
      GO TO 104
  111 IAX=(IL-1)/2
      MA(I,IL)=SAUV(IAX)
      IL=IL+1
      MA(I,IL)=KK
      KK=KK+1
      IL=IL+1
      IF(IL.LE.NN)GO TO 123
      GO TO 104
  106 IF(SAUV(1).NE.0)GO TO 112
  128 IF(MA(I,2*J+1).EQ.0) GO TO 125
      J=J+1
      GO TO 128
  125 IF(J.EQ.NA)GO TO 104
      MA(I,1)=I
      MA(I,2)=1
      MA(I,2*J+1)=GARD(2,J)
      MA(I,2*J+2)=1
      J=J+1
      GO TO 210
  112 MA(I,1)=I
      MA(I,2)=1
      IK=1
      JK=J+1
  114 MA(I,2*JK-1)=SAUV(IK)
      IF(MA(I,2*JK).EQ.0)MA(I,2*JK)=IK
      JK=JK+1
      IK=IK+1
      IF(SAUV(IK).NE.0)GO TO 114
      J=J+IK-1
      IF(J.EQ.NA)GO TO 104
      GO TO 113
  124 MA(I,2*J+1)=GARD(2,J)
      MA(I,2*J+2)=1
  104 CONTINUE
      CALL TRACE(PO,NA)
```

```
      WRITE(LW,4001)
 4001 FORMAT(1X,'MATRICE MA')
      DO 600 I8=1,NA
      I9=1
      N8=NN-1
      DO 601 J8=1,N8,2
      ECRMAT(I9)=MA(I8,J8)
      I9=I9+1
  601 CONTINUE
      WRITE(LW,4000) (ECRMAT(I9),I9=1,NA)
 4000 FORMAT(' ',40I3)
  600 CONTINUE
      DO 130 I=1,NN
  130 PO2(I)=PO(I)
      READ (LR,2000)NB
      NN1=2*NB
      READ (LR,2001)(PO(I),I=1,NN1)
      CALL COORD(MB,NB,PO)
      CALL TRACE(PO,NB)
      WRITE(LW,4002)
 4002 FORMAT(1X,'MATRICE MB')
      DO 602 I8=1,NB
      I9=1
      DO 603 J8=1,NB
      ECRMAT(I9)=MB(I8,J8)
      I9=I9+1
  603 CONTINUE
      WRITE(LW,4000)(ECRMAT(I9),I9=1,NB)
  602 CONTINUE
 1001 FORMAT(40I2)
      ILJ=0
C     CONSTRUCTION DU TABLEAU D'AFFECTATION
      DO 90 NABB=1,NA
      NAB=NA-NABB+1
      DO 1 I=1,20
      DO 1 J=1,20
      ICOMP(I,J)=0
      ICOMK(I,J)=0
    1 CONTINUE
      NIB=0
      DO 3 I=1,NAB
      IAX=NAB-NA+I
      MINK=MAXO(1,IAX)
      IAX=NB-NAB+I
      MAXK=MINO(NB,IAX)
      DO 4 K=MINK,MAXK
      NBR=0
      DO 5 MU=1,NA
      IF(MU.EQ.I)GO TO 5
      NU=1
    6 IF(MA(MU,NU).EQ.MA(I,1))GO TO 7
      NU=NU+2
      GO TO 6
    7 IF(MU.LT.I)GO TO 8
      IAX1=K+1
```

```
      IAX=NAB-NA+MU
      MINT=MAXO(IAX1,IAX)
      IAX=NB-NAB+MU
      MAXT=MINO(NB,IAX)
      GO TO 9
    8 IAX=NAB-NA+MU
      MINT=MAXO(1,IAX)
      IAX1=K-1
      IAX=NB-NAB+MU
      MAXT=MINO(IAX1,IAX)
    9 IAX=NAB-NA+(NU+1)/2
      IAX1=NB-NAB+(NU+1)/2
      MINS=MAXO(2,IAX)
      MAXS=MINO(NB,IAX1)
      NU=NU+1
      IND=MA(MU,NU)
      IF(IND.GT.1)MINS=MINS-IND+1
      ICO=1
   11 NU=NU+2
      IF(MA(MU,NU).LE.ICO)GO TO 10
      ICO=ICO+1
      IF(NU.LE.NN)GO TO 11
   10 MAXS=MAXS+ICO-1
      DO 35 L=MINT,MAXT
      DO 35 M=MINS,MAXS
      IF(MB(L,M).NE.MB(K,1))GO TO 35
      NBR=NBR+1
      GO TO 5
   35 CONTINUE
    5 CONTINUE
      IF(NBR.GE.(NAB-1))ICOMP(I,K)=1
    4 CONTINUE
    3 CONTINUE
      KK=0
      IKK=0
      LI=1
      ICC=0
      LJ=1
C     VALIDATION AFFECT PAR AFFECT AVEC G**-1
   14 IF(ICOMP(LI,LJ).EQ.1)GO TO 12
      LJ=LJ+1
      IF((NB-LJ).LT.(NAB-LI))GO TO 13
      IF(LJ.EQ.(NB+1))GO TO 13
      GO TO 14
   13 DO 15 I=1,NB
   15 ICOMK(LI,I)=0
      NX=2
   88 LI=LI+1
      IF(LI.EQ.(NA+1)) GO TO 17
      LZ=LI-NX
      LQ=1
   87 IF(ICOMK(LZ,LQ).EQ.1)GO TO 85
      LQ=LQ+1
      IF(LQ.EQ.(NB+1))GO TO 86
      GO TO 87
```

```
   86 NX=NX+1
      GO TO 88
   85 LJ=LQ+1
      GO TO 14
   17 LI=LI-1
      IF(LI.EQ.0)GO TO 500
      LA=1
   18 IF(ICOMK(LI,LA).EQ.1)GO TO 16
      LA=LA+1
      IF(LA.EQ.(NB+1))GO TO 17
      GO TO 18
   16 ICOMK(LI,LA)=0
      ICC=ICC-1
      LJ=LA+1
      IF(LJ.EQ.(NB+1))GO TO 17
      GO TO 14
   12 IF((NB-LJ).LT.(NAB-LI))GO TO 13
      ICOMK(LI,LJ)=1
      ICC=ICC+1
      IF(ICC.EQ.NAB)GO TO 19
      LI=LI+1
      LJ=LJ+1
      IF(LJ.EQ.(NB+1))GO TO 17
      IF(LI.EQ.(NA+1))GO TO 17
      GO TO 14
   19 KK=KK+1
C     MISE SOUS FORME (I,J) DE MA
      NAA=2*NA-1
      DO 807 IA=1,NA
      DO 808 JA=1,NAA,2
      JAA=(JA+1)/2
      MA1(IA,JAA)=MA(IA,JA)
  808 CONTINUE
  807 CONTINUE
C     REMPLISSAGE DE AVIRER POUR MA
      DO 800 IA=1,NB
      AVIRER(IA)=0
  800 CONTINUE
      DO 801 IA=1,NAB
      AVIRER(IA)=1
  801 CONTINUE
      CALL SMAT(MA1,NA,SMA,AVIRER)
C     REMPLISSAGE DE AVIRER POUR MB
      DO 802 IB=1,NB
      AVIRER(IB)=0
  802 CONTINUE
      DO 803 IB=1,NB
C     RECHERCHE SI IB EST CONCERNE PAR L'AFFECT
      DO 804 IBB=1,NAB
      IF(ICOMK(IBB,IB).EQ.0)GOTO  804
      AVIRER(IB)=1
      GOTO 803
  804 CONTINUE
  803 CONTINUE
      CALL SMAT(MB,NB,SMB,AVIRER)
```

```
C       COMPARAISON DES SOUS MATRICES SMA ET SMB
        DO 810 I8=1,NAB
        DO 811 J8=1,NAB
        ECRMAT(J8)=SMA(I8,J8)
  811   CONTINUE
        ECRMAT(NAB+1)=0
        DO 812 J8=1,NAB
        J9=J8+NAB+1
        ECRMAT(J9)=SMB(I8,J8)
  812   CONTINUE
  810   CONTINUE
C       TRANSFORMATION DE SMA EN FONCTION DE L'AFFECTATION
        DO 350 I=1,NAB
        DO 351 J=1,NB
        IF(ICOMK(I,J).NE.1) GO TO 351
        JEGAL(I,1)=I
        JEGAL(I,2)=J
  351   CONTINUE
  350   CONTINUE
        DO 900 I=1,NAB
        DO 951 J=1,NAB
        SMA(I,J)=JEGAL(SMA(I,J),2)
  951   CONTINUE
  900   CONTINUE
        DO 805 IC=1,NAB
        DO 806 JC=1,NAB
        IF(SMA(IC,JC).EQ.SMB(IC,JC)) GOTO 806
        IKK=IKK+1
        GOTO 140
  806   CONTINUE
  805   CONTINUE
  140   ICOMK(LI,LJ)=0
        ICC=ICC-1
        IF(LI.EQ.NA)GO TO 23
        IF(LJ.EQ.NB)GO TO 24
        LJ=LJ+1
   25   IF(ICOMP(LI,LJ).EQ.1)GO TO 12
        LJ=LJ+1
        IF(LJ.NE.(NB+1))GO TO 25
   24   DO 26 I=1,NB
   26   ICOMK(LI,I)=0
        LI=LI+1
        II=LI-2
        DO 27 I=1,NB
        IF(ICOMK(II,I).NE.1)GO TO 27
        LJ=I
        GO TO 28
   27   CONTINUE
   28   LJ=LJ+1
        GO TO 14
   23   LJ=LJ+1
        IF(LJ.NE.(NB+1))GO TO 14
        GO TO 17
  500   KKK=KK-IKK
        WRITE(LW,1011) KKK,NAB,IKK
```

```
      ILJ=ILJ+1
      ANAB=NAB*9
      PO3(ILJ)=ANAB
      ILJ=ILJ+1
      LKK=KKK/5
      PO3(ILJ)=LKK
 1011 FORMAT(1X,I5,' AFFECTATIONS POSSIBLES POUR NAB=',I3,'  IL Y EN A E
     1U ',I5,' ELIMINEES PAR G**-1')
   90 CONTINUE
      ILJ=ILJ/2
      CALL TRACE(PO3,ILJ)
      INDIC=INDIC+1
      STOP
      END
      SUBROUTINE TRACE(TAB,N)
      INTEGER TAB(60),LIGNE(120),AMIN,AUX
      INTEGER NU
      DATA JA/4H-   /,JB/4HI   /,JP/4H*   /,JBB/4H    /
      INTEGER CH(10)
      DATA CH/'0','1','2','3','4','5','6','7','8','9'/
      LR=5
      LW=6
      NN=2*N
  100 FORMAT(' ',60A1)
  101 FORMAT(' ',60A1)
      DO 200 I=2,NN,2
      TAB(I-1)=TAB(I-1)+10
      TAB(I-1)=TAB(I-1)/2
      TAB(I)=119-TAB(I)
      IF(TAB(I).LE.0) TAB(I)=1
  200 TAB(I)=TAB(I)*25/60
      DO 11 I=1,60
   11 LIGNE(I)=JA
      LIGNE(1)=JB
      LIGNE(60)=JB
      WRITE(LW,101)(LIGNE(I),I=1,60)
      ISAV=0
      L=1
   15 DO 1 I=1,60
    1 LIGNE(I)=JBB
      LIGNE(1)=JB
      LIGNE(60)=JB
      AMIN=1000
      IFLAG=0
      DO 2 I=2,NN,2
      IF(TAB(I)-AMIN)30,2,2
   30 AMIN=TAB(I)
      IFLAG=1
      II=I
    2 CONTINUE
      IF(IFLAG)20,20,9
    9 TAB(II)=1000
      J=AMIN
      LJ=II-1
      IK=J-ISAV-1
```

```
      ISAV=J
      IF(IK)13,13,35
   35 L=L+IK
      DO 10 K=1,IK
      DO 14 JK=1,60
      LIGNE(JK)=JBB
   14 LIGNE(1)=JB
      LIGNE(60)=JB
   10 WRITE(LW,100)(LIGNE(I),I=1,60)
   13 JJ=TAB(LJ)
      LIGNE(JJ)=JP
      NU=(LJ+1)/20
      IJJ=JJ+1
      NUP=NU+1
      LIGNE(IJJ)=CH(NUP)
      NU=((LJ+1)/2)-NU*10
      NUP=NU+1
      IJJ=IJJ+1
      LIGNE(IJJ)=CH(NUP)
      DO 7 K=2,NN,2
      AUX=AMIN-TAB(K)
      IF(AUX.EQ.0) GOTO 8
    7 CONTINUE
      GO TO 12
    8 II=K
      GO TO 9
   12 WRITE(LW,100)(LIGNE(I),I=1,60)
      L=L+1
      GO TO 15
   20 IA=25-L
      DO 21 IB=1,IA
      DO 22 I2=1,60
      LIGNE(I2)=JBB
      LIGNE(60)=JB
   22 LIGNE(1)=JB
      WRITE(LW,100)(LIGNE(I),I=1,60)
   21 CONTINUE
      DO 23 I=1,60
   23 LIGNE(I)=JA
      WRITE(LW,100)(LIGNE(I),I=1,60)
      DO 24 I=1,60
   24 LIGNE(I)=JBB
      WRITE(LW,101)(LIGNE(I),I=1,60)
      RETURN
      END
      SUBROUTINE COORD(MOT,N,PO)
      INTEGER   IDZON(30),PO(60),MOT(20,20),TAB(60),MB(20,20)
      INTEGER COU(30,2),GARD(30),DIST(30),DIST2(30),ZMAX,ZONE
C     *** LECTURE COORDONNEES ***
      LR=5
      LW=6
C     *** CALCULL DES DISTANCES***
      DO 10 I=1,N
      DO 20 J=1,N
      IF(J-I)1,2,1
```

```
    1 II=2*I
      JJ=2*J
      MI=II-1
      MJ=JJ-1
      DIST(J)=(PO(MI)-PO(MJ))**2+(PO(II)-PO(JJ))**2
      DIST2(J)=DIST(J)
      GO TO 20
    2 DIST(J)=1000000
      DIST2(J)=DIST(J)
   20 CONTINUE
      MOT(I,1)=I
      L=2
   30 M=1
      MIN=1000000
      DO 40 K=1,N
      IF(DIST(K)-1000000)11,40,500
   11 IF(MIN-DIST(K))40,4,5
    4 M=M+1
      GO TO 16
    5 IF(M-1)500,15,15
   15 M=1
   16 MIN=DIST(K)
      GARD(M)=K
      DIST(K)=1000000
   40 CONTINUE
      IF(M-1)500,6,7
    6 MOT(I,L)=GARD(M)
      DIST2(GARD(M))=1000000
      GO TO 8
    7 DO 50 IM=1,M
      IA=IM-1
      IL=IA+L
      IF(IM-1)500,18,17
   18 MOT(I,IL)=GARD(1)
      DIST2(GARD(1))=1000000
      GO TO 50
   17 MOT(I,IL)=-GARD(IM)
      DIST2(GARD(IM))=1000000
   50 CONTINUE
    8 L=L+M
      DO 60 LL=1,N
   60 DIST(LL)=DIST2(LL)
      IF(L-N)30,30,10
   10 CONTINUE
C     *** IMPRESSIONS ***
      K=3
      GO TO 502
  500 WRITE(LW,1003)
 1003 FORMAT(1X,'ERREUR')
      RETURN
  502 DO 70 I=1,N
      DO 70 J=1,N
   70 MOT(I,J)=IABS(MOT(I,J))
      RETURN
      END
```

```
      SUBROUTINE SMAT(MAT,N,SMA,VIR)
      INTEGER MAT(20,20),SMA(20,20),VIR(20)
C     SPRG CONSISTANT A EXTRAIRE UNE SOUS MATRICE SMA D'UNE MATRICR
C     MAT EN ELIMINANT DE MAT LES PTS I TELS QUE VIR(I)=0
      IS=1
      DO 1 I=1,N
      IF(VIR(MAT(I,1)).EQ.0)GOTO 1
      JS=1
      DO 2 J=1,N
      IF(VIR(MAT(I,J)).EQ.0)GOTO 2
      SMA(IS,JS)=MAT(I,J)
      JS=JS+1
    2 CONTINUE
      IS=IS+1
    1 CONTINUE
      RETURN
      END
/*
//GO.SYSIN DD *
10
  10  15  40  22  18  25  44  30  70  35  45  45  10  50  40  60  80  75  90
12
  10  15  40  22  18  25  44  30  70  35  45  45  30  50  12  55  60  70  90
  90  80  40  90
/*
```

ANNEXE 2

Matrice parenthésée - 9 points.

1	1	2	1	3	1	4	1	6	1	8	1	5	1	7	1	9	1
2	1	3	1	1	1	4	1	6	1	8	1	5	1	7	1	9	1
3	1	2	1	4	1	1	1	5	1	6	1	8	1	7	1	9	1
4	1	3	1	5	1	6	1	8	1	2	1	1	1	7	1	9	1
7	1	6	1	4	1	5	1	3	1	8	1	2	1	1	1	9	1
6	1	7	1	4	1	5	1	3	1	8	1	2	1	1	1	9	1
8	1	4	1	9	1	5	1	3	1	2	1	7	1	6	1	1	1
9	1	8	1	4	1	5	1	3	1	2	1	6	1	1	1	7	1
5	1	4	1	6	1	8	1	3	1	2	1	7	1	9	1	1	1

Matrice catalogue - 11 points.

1	4	2	7	3	5	6	8	10	9	11
2	3	1	4	5	7	6	8	9	10	11
3	5	6	2	4	1	7	8	9	10	11
4	1	7	2	8	5	3	6	10	9	11
5	3	6	2	8	7	4	1	9	10	11
6	5	3	8	9	7	10	11	2	4	1
7	8	4	6	10	5	9	1	2	3	11
8	7	10	9	6	4	11	5	3	2	1
9	10	11	8	6	7	5	4	3	2	1
10	9	11	8	6	7	5	4	3	2	1
11	10	9	8	6	7	5	4	3	2	1

On prend toutes les affectations possibles : pour 4 points par exemple, le combinatoire donne $C_q^4 \times C_{11}^4$. Pour des raisons de temps on s'est contenté de regarder dans un premier temps (avant de faire une étude complète sur un partiel particulier) si des affectations étaient possibles ou non (par construction du tableau COM). Il apparaît qu'à 8 points et 9 points, il n'y a plus d'affectations possibles.

	1	4	2	7	3	5	6	8	10	9	11
1	1	1	1	1	1	1	1	1	0	0	0
2	1	1	1	1	1	1	1	1	1	0	0
3	1	1	1	1	1	1	1	1	1	1	0
4	1	1	1	1	1	1	1	1	1	1	1
7	1	1	1	1	1	1	1	1	1	1	1
6	1	1	1	1	1	1	1	1	1	1	1
8	0	1	1	1	1	1	1	1	1	1	1
9	0	0	1	1	1	1	1	1	1	1	1
5	0	0	0	1	1	1	1	1	1	1	1

AFFECTATION POSSIBLE POUR NAB # 4

1	1	1	1	1	1	1	0	0	0	0
1	1	1	1	1	1	1	1	0	0	0
1	1	1	1	1	1	1	1	1	0	0
1	1	1	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1	1	1
0	1	1	1	1	1	1	1	1	1	1
0	0	1	1	1	1	1	1	1	1	1
0	0	0	1	1	1	1	1	1	1	1
0	0	0	0	1	1	1	1	1	1	1

AFFECTATION POSSIBLE POUR NAB # 5

1	1	1	1	1	1	0	0	0	0	0
1	1	1	1	1	1	1	0	0	0	0
1	1	1	1	1	1	1	1	0	0	0
1	1	1	1	1	1	1	1	1	0	0
0	1	1	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	1	1	1	1
0	0	0	1	1	1	1	1	1	1	1
0	0	0	0	1	1	1	1	1	1	1
0	0	0	0	0	1	1	1	1	1	1

AFFECTATION POSSIBLE POUR NAB # 6

1	1	1	1	1	0	0	0	0	0	0
1	1	1	1	1	1	0	0	0	0	0
1	1	1	1	1	1	1	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0
0	0	1	1	1	1	1	1	1	0	0
0	0	0	1	1	1	1	1	1	1	0
0	0	0	0	1	1	1	1	1	1	1
0	0	0	0	0	1	1	1	1	1	1
0	0	0	0	0	0	1	1	1	1	1

AFFECTATION POSSIBLE POUR NAB # 7

1	1	1	1	0	0	0	0	0	0	0
1	1	1	1	1		0	0	0	0	0
0	1	1	1	1	1	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0	0
0	0	0	1	0	0	1	0	0	0	0
0	0	0	0	1	0	1	0	0	0	0
0	0	0	0	0	1	1	1	1	1	0
0	0	0	0	0	0	0	0	1	1	1
0	0	0	0	0	0	0	1	1	1	1

ZERO AFFECTATION POSSIBLE POUR NAB # 8

1	1	1	0	0	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0

ZERO AFFECTATION POSSIBLE POUR NAB # 9

En prenant un partiel de P points particulier on obtient les résultats :

P	9	8	7	6	5	4
1 partiel particulier	0	0	155	462	462	330
1 partiel particulier	0	0	130	462	462	330
Résultat statistique	0	0	affec	affec	affec	affec

Le terme C_m^p étant un paramètre indépendant des figures, nous ne considérons que l'écart entre C_n^p (matrice catalogue au hasard) et le nombre d'affectations réel pour un partiel de p points (matrice catalogue représentant une figure).

§5 UN OPERATEUR $\mathcal{O}(X)$ INVARIANT DANS LES PROJECTIONS PLANES

5.1. Définition de $\mathcal{O}(X)$

Reprenons le formalisme défini au §1 de ce chapitre et soit $X = \{A_0, A_1, \ldots, A_n\}$ une forme constituée par n primitives.

Nous considérons l'une des primitives comme origine, A_0 par exemple, Alors $A_i\ A_0$ définit une direction. En partant de cette direction, un vecteur d'origine A_i et tournant dans le sens direct rencontrera les autres points A_j de X dans un certain ordre.

Nous appelons u_i l'ensemble ordonné ainsi obtenu :

$$u_i = A_i\ A_{j_1}\ A_{j_2} \ldots A_{j_{n-1}}$$

Nous pouvons répéter l'opération en chaque point de la forme X à l'exception de A_0. Chaque fois, un n-mot u_j de n lettres est obtenu et on désigne par U_X l'ensemble des n-mots correspondant à la forme X.

Une projection du plan π sur le plan π' transforme le mot $A_1\ A_2 \ldots A_n$ en le mot $A'_1\ A'_2 \ldots A'_n$. Il est facile de vérifier que l'ensemble de n-mots U'_X obtenu à partir de la transformée X' de X, se déduit de U_X en substituant A'_j à A_j.

Ainsi l'opérateur $\mathcal{O}(X)$ qui fait passer de X à U_X est un algorithme invariant dans les projections planes. On peut aussi vérifier qu'il est invariant dans les similitudes planes.

Remarques :

1. Supposons que la projection plane considérée soit une "projection parallèle" c'est-à-dire dont le centre est à l'infini. Soit G le centre de gravité des points A_i qui peuvent être pondérés par les valeurs numériques z_i.

$$\sum_i z_i\ \overrightarrow{GA_i} = 0$$

La projection G' de G est encore le centre de gravité de la forme projetée X'. Ainsi, pour une projection parallèle, on peut prendre G comme point de référence A_0.

2. Supposons que A_0 soit à l'infini. Cela signifie que la direction de départ de chaque exploration est la même pour tous les points A_i. Considérons une projection plane qui conserve le point A_0 à l'infini. Ce qui revient à supposer que A_0 se trouve à l'infini sur la droite d'intersection des deux plans considérés. Pour le nouveau plan, la direction de départ du vecteur de rotation est la même pour tous les points. C'est la projection de la direction de départ dans le plan précédent. Nous nous limiterons à ce cas particulier dans les exemples d'applications que nous proposons.

5.2 Application pratique aux images

Prenons le demi axe positif des ordonnées comme direction de départ. Le sens de rotation est celui des aiguilles d'une montre.

Soit $\{A_1\ A_2\ \ldots\ A_n\}$ l'alphabet des noms des primitives de l'image considérée. Le résultat partiel obtenu à partir du point A_i s'écrit :

$$u_i = \underbrace{A_i\ A_k\ A_e\ \ldots\ A_n}_{\text{n éléments}}$$

n mots tels que u_i forme une matrice n x n : le "résultat U_X" de l'opérateur.

Toute fonction monotone croissante de l'angle permet d'obtenir les u_i. Soit q la taille du registre dans lequel est enregistré l'angle de $A_i\ A_j$ avec la direction horizontale de départ. Soient x et y les projections de $A_i\ A_j$ sur les axes horizontal et vertical. Il est intéressant d'utiliser alors la fonction $\varphi(A_j \mid A_i)$ suivante :

$$0, \frac{\pi}{2}, \ldots \left[2 + \frac{|y|}{|x| + |y|}\right] 2^q - 2$$

$$\frac{\pi}{2}, \pi, \ldots \left[3 + \frac{|y|}{|x| + |y|}\right] 2^q - 2$$

$$\pi, \frac{3\pi}{2}, \ldots \left[\frac{|y|}{|x| + |y|}\right] 2^q - 2$$

$$\frac{3\pi}{2}, 2\pi, \ldots \left[1 + \frac{|y|}{|x| + |y|}\right] 2^q - 2$$

Une telle fonction φ permet d'ordonner les angles avec une très grande précision.

N. B. : on verra plus loin qu'il peut être utile de décider que deux points A_k et A_e sont vus "en ligne" à partir d'un troisième A_i. Autrement dit, que les angles des vecteurs $A_i\ A_k$ et $A_i\ A_e$ sont les mêmes par rapport à la direction de référence. Ceci sera possible en jouant sur la précision de la fonction φ.

Les figures IV.13.X et IV.13.Y montrent des images X et Y de 8 points : A, B ... H. Les résultats correspondants sont donnés par les figures IV.14.X et IV.14.Y.

L'image Y présente des points alignés. Ceci est noté dans la matrice 8 x 8 correspondante. Par exemple, si E et C sont alignés avec A, on remarque la présence du chiffre 1 après E et 2 après C. De même C et A sont alignés avec E...

Remarques :

1. Une modification de la direction origine pour les rotations transforme généralement U_X ; il peut se produire une permutation des (n - 1) dernières lettres d'une ligne.

Une rotation de l'image dans le plan peut introduire de telles permutations. L'opérateur envisagé n'est donc pas invariant par rapport à la rotation. Cependant, il est préférable à celui plus général déjà introduit car lorsqu'on compare deux figures inconnues, la correspondance entre les points A_0 de chacune d'elles doit être connue.

En général, ceci n'est pas possible.

On pourrait prendre par exemple comme point A_0 le centre de gravité, mais cela réduit les projections planes aux projections parallèles. D'autre part, en utilisant la direction parallèle aux abscisses comme origine de la rotation on peut comparer des figures ayant un nombre différent de points primitifs et aussi on peut déduire l'enveloppe convexe. Quoiqu'il en soit l'invariance par rapport à la rotation peut être très utile pour certains problèmes et ce procédé mérite d'être également étudié.

2. Si A_0 est à l'infini, l'image peut être parcourue dans le même sens que celui du vecteur rotation. Le "balayage" est fait selon une ligne horizontale, l'ordonnée de cette ligne allant de $+\infty$ à $-\infty$. Ainsi les n points caractéristiques de l'usage sont pris dans un certain ordre. Nous adoptons cet ordre pour la première colonne de la matrice U_X, qui est alors définie de façon unique (cf. fig. IV.14.X et IV.14.Y).

5.3 Quelques propriétés déduites de U_X

La matrice U_X, invariante pour les projections planes qui conserve la direction horizontale, est utilisée pour reconnaître des classes de forme assez grandes. Auparavent, on peut déduire de U_X seule quelques propriétés de l'image analysée. Par exemple, il est possible d'établir le sens des triangles ou de décider si un quadrilatère formé par quatre des points de la figure est convexe ou non.

L'ENVELOPPE CONVEXE

Il est très intéressant de pouvoir trouver l'enveloppe convexe d'une image c'est-à-dire les points sur lesquels un polygone convexe peut être construit de telle manière que tous les autres points de l'image soient à l'intérieur de ce polygone.

Soient deux points A_k et A_e tels que, dans U_X :

a. A la ligne commençant par A_k , appelée ligne A_k, le point immédiatement après A_k est A_e.

b. La ligne A_e se termine par A_k.

On montre facilement que A_k et A_e appartiennent à l'enveloppe convexe, et que A_e suit immédiatement A_k sur cette enveloppe si elle est parcourue dans le sens adopté pour la rotation.

Le point A_m suivant sur l'enveloppe convexe est le point qui suit A_k dans la ligne A_e. Si A_k est le dernier point de cette ligne, A_m en est le second (permutation circulaire).

L'algorithme s'arrête lorsqu'on a trouvé à nouveau le point A_k comme point suivant.

Par exemple, en revenant à l'exemple de la figure IV.13.X dont la matrice U_X est donnée par la figure IV.14.X, il est facile d'obtenir, grâce à cet algorithme, les enveloppes convexes des figures partielles extraites de cette figure de 8 points. Les résultats sont donnés par la figure IV.15 : une ligne présente les points de la figure partielle, la ligne au-dessous et décallée donne les points de l'enveloppe convexe correspondante. Pour l'image totale A, B ... H, l'enveloppe et C A B G H F.

RECONSTITUTION DE L'IMAGE A PARTIR DE U_X

La matrice U_X étant donnée (résultat de l'opérateur $\mathcal{O}$) est-il possible de réobtenir l'image de départ X ?

Rappelons que l'intérêt de la représentation U_X n'est pas une réduction de l'information mais qu'une même matrice U_X représente une grande classe de formes, modulo une transformation géométrique Θ.

On peut dire également qu'une "déformation" de l'image ne modifie pas U_X. En effet, considérons une forme X constituée par n points, les n(n - 1) segments qui joignent deux points A_i A_j ainsi que les n demi-droites horizontales qui joignent un point A_i au point A_0 à l'infini.

Si aucune de ces lignes ne passe sur un des points de l'image lors d'une déformation Δ de celle-ci, alors la matrice U_X reste inchangée. Une telle transformation Δ peut altérer considérablement la configuration de l'image de départ. Ainsi, toute image reconstituée à partir de U_X peut être très différente de l'image originale X.

On a vu que l'enveloppe convexe d'une sous-figure quelconque d'une figure donnée peut être déduite de U_X. On utilise cette propriété pour reconstruire X à partir de U_X. Les points appartenant à l'enveloppe convexe sont disposés sur un cercle dans l'ordre où ils doivent être sur l'enveloppe. La matrice U_X permet de déterminer les régions, définis par plusieurs demi-droites du plan, où les autres points doivent se trouver. L'ensemble de ces autres points est rangé dans l'ordre des degrés de liberté décroissants. Les premiers dans cet ordre, c'est-à-dire ceux qui ont le moins de contraintes, sont placés aux barycentres des régions qui leur sont affectées grâce à U_X.

Il peut se produire une impossibilité avant d'avoir épuisé la totalité des points de l'ensemble.

Le procédé est réitéré après avoir modifié la position des points de l'enveloppe de façon aléatoire.

Bien que nous soyons persuadé que ceci n'est pas la meilleure façon de procéder, les itérations successives permettent généralement d'augmenter le nombre de points qui peuvent être replacés conformément à U_X.

La figure IV.17 représente une reconstitution de la figure IV.16. Tous les points excepté le point F ont été replacés sur un total de 13 points. L'enveloppe convexe est A C J M I B. La matrice U_X ne donne aucune indication sur la forme de l'enveloppe, ce qui conduit à une importante déformation.

La figure IV.20 donne la reconstitution de la figure IV.19 à partir de U_X. Sur les 38 points, 30 ont été restitués. Ce nombre paraît être une limite.

Bien qu'il soit intéressant de pouvoir restituer une image à partir de la représentation par U_X, ce n'est pas là l'objet de ce procédé, qui doit principalement permettre la reconnaissance de l'image. Cependant, ces expériences ont permis de montrer à quel point deux images ayant la même représentation U_X peuvent être différentes.

5.4. Identification d'une forme à partir de U_X

La comparaison de deux formes X et Y ayant le même nombre n de points caractéristiques, est tout-à-fait simple.

Soient $A_1\ A_2 \ldots A_n$ les éléments de X

$B_1\ B_2 \ldots B_n$ ceux de Y.

Les deux formes X et Y sont "équivalentes", c'est-à-dire peuvent être déduites l'une de l'autre par une transformation Θ et éventuellement une déformation Δ, s'il existe une application bijective Γ :

$$\Gamma : A_i \rightarrow B_i$$

telle que U_X soit transformée en U_Y et inversement :

$$\Gamma^{-1} : B_j \rightarrow A_i$$

L'application est définie en comparant la première colonne de U_X à celle de U_Y. Puis la vérification des correspondances est effectuée sur les lignes des matrices. Dans le cas des figures IV.14.X et IV.14.Y, par exemple, les applications Γ et Γ^{-1} sont :

$\Gamma_{\uparrow}$, $\Gamma^{-1}{}_{\downarrow}$ A C B D F E G H (forme X)
A B D C F E G H (forme Y)

Les deuxièmes lignes de U_X et U_Y sont :

C A B D G E H F
B A G E H F C D

En utilisant Γ, une correspondance totale ne peut pas être établie entre ces deux lignes. On décide alors que les deux formes ne sont pas équivalentes.

Si des points sont "alignés", ils peuvent permuter entre eux pendant l'identification.

La discrimination entre deux formes de n points peut être très grande. Elle dépend naturellement de la précision angulaire retenue, précision qui peut être réduite : deux points A_i et A_j d'une forme sont "alignés" avec A_k si l'angle $(A_i A_k, A_i A_j)$ est inférieur à un ϵ donné. Ainsi des classes de points peuvent apparaître dans chaque ligne de U_X ou U_Y. Les points d'une même classe peuvent permuter. Un procédé analogue a été utilisé avec l'opérateur sur les distances (cf. §5.3 et [ ]). Nous l'appelons le parenthésage.

DETECTION DE SOUS FIGURES EQUIVALENTES

Soient deux formes X et Y, $|X|$ et $|Y|$ le nombre de points caractéristiques de chacune d'elles. L'opérateur linguistique $\mathcal{O}$ fournit U_X et U_Y à partir desquels il est possible de trouver deux sous figures Z et Z' telles que $Z \subseteq X$ et $Z' \subseteq Y$, $|Z| = |Z'|$ et une application $\Gamma : U_Z \to U_{Z'}$, $\Gamma^{-1} : U_{Z'} \to U_Z$.

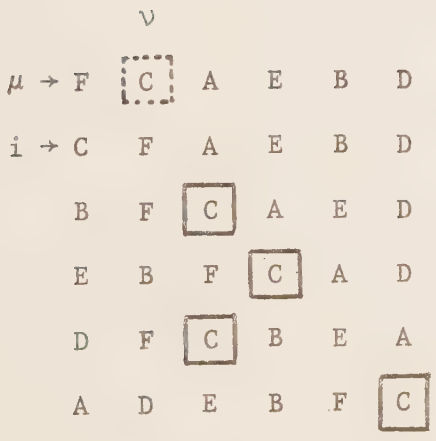
En d'autres termes, en supprimant un certain nombre de points de chaque forme il est possible de trouver deux ensembles de points satisfaisant l'algorithme de correspondance précédent.

Nous allons montrer le déroulement du processus sur un exemple : soit X une forme de 6 points (fig. IV.18.X) et Y une forme de 8 points (fig.IV.18.Y). On essaie tout d'abord de voir si X est entièrement contenue dans Y (soit X = Z).

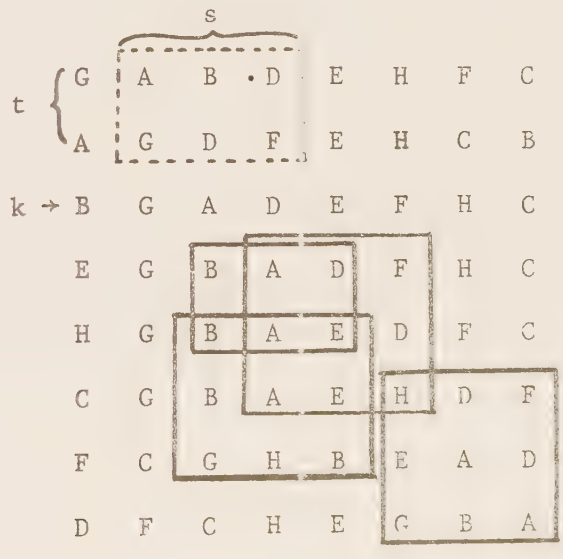
La correspondance entre certains points apparaît parfois impossible a priori. Par exemple, F_X et D_Y. D'autre part, si on cherche les points de Y qui pourraient correspondre à C_X on ne trouve que A_Y B_Y et E_Y si on considère la première colonne.

Cependant, si $\Gamma : C \to B$, B devra se trouver dans les cinq domaines rectangulaire de U_Y dessinés plus bas : la position de B dans U_Y doit être voisine de celle de C dans U_X. Dans l'exemple choisi, B ne figure que dans 4 domaines sur les 5 existant.

B_Y ne peut donc pas être affecté à C_X par l'application Γ.

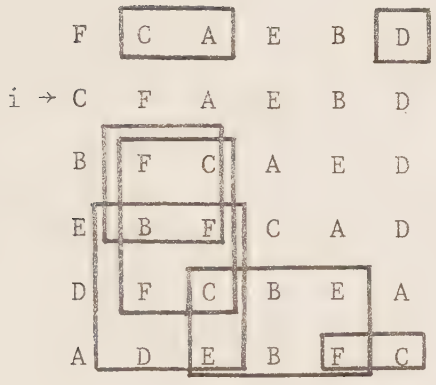


U_X

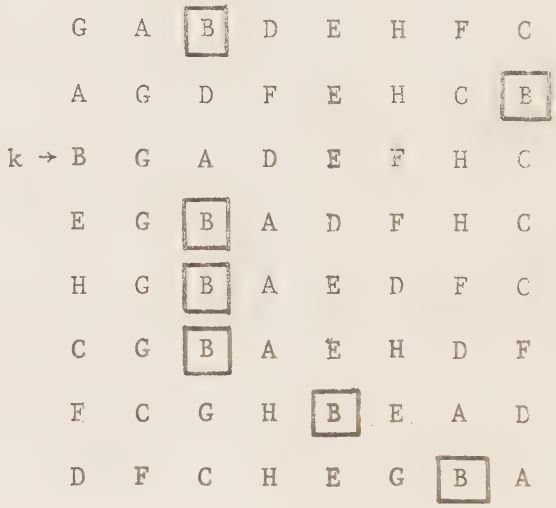


U_Y

Dans l'application Γ^{-1} : B → C, le point C devra se trouver dans cinq des sept domaines rectangulaires dessinés sur le tableau de la figure U_X ci-dessous. C'est bien le cas puisqu'on retrouve C dans six de ces domaines.



U_X



U_Y

Donnons alors les conditions générales de correspondance et soit $U_X(i, j)$ un élément de la matrice U_X, $U_Y(k, 1)$ un élément de U_Y.

1. Correspondance de la 1ère colonne :

Si $$U_X(i, 1) \rightarrow U_Y(k, 1)$$

$$\text{Sup}(1, |Z| - |X| + i) \leqslant k \leqslant \text{Inf}(|Y|, |Y| - |Z| + i)$$

(Z est le nombre de points du partiel possible).

2. Un élément X peut se retrouver dans plusieurs positions dans U_X. Soient μ et ν ses coordonnées, telles que $U_X(i, 1) = U_X(\mu, \nu)$. Chaque point détermine un rectangle dans U_Y défini par t et s.

Les conditions sur t sont les suivantes :

pour $< i$ $\quad \sup(1, |Z| - |X| + \mu) \leqslant t \leqslant \inf(k - 1, |Y| - |Z| + \mu)$

pour $> i$ $\quad \sup(k + 1, |Z| - |X| + \mu) \leqslant t \leqslant \inf(|Y|, |Y| - |Z| + \mu)$

et s peut satisfaire la double condition :

$$\sup(2, |Z| - |X| + \nu) \leqslant s \leqslant \inf(|Y|; |Y| - |Z| + \nu)$$

$|X| - 1$ domaines de ce type sont ainsi définis dans U_Y. L'application $\Gamma : U_X(i, 1) \rightarrow U_Y(k, 1)$ est possible si $U_Y(k, 1)$ est au moins dans $|Z| - 1$ de ces domaines. On peut construire pour la commodité, des matrices d'affectation M_Z. Pour l'exemple traite la matrice d'affectation est la suivante :

X \ Y	G	A	B	E	H	C	F	D
F	0	1	1	0	0	0	0	0
C	0	1	0	0	0	0	0	0
B	0	0	0	0	0	0	0	0
E	0	0	0	1	1	0	0	0
D	0	0	0	0	0	1	1	0
A	0	0	0	0	0	0	1	1

La présence d'un 0 dans une case signifie qu'au moins une des affectations est impossible. Si on met un 1 c'est que les deux sont possibles en accord avec la règle précédente.

Dans notre exemple, l'affectation n'est pas possible pour toute figure partielle Z telle que $|Z| = 6$.

X \ Y	G	A	B	E	H	C	F	D
F	1	1	1	1	0	0	0	0
C	1	1	1	1	0	0	0	0
B	0	0	1	1	1	1	0	0
E	0	0	1	1	1	1	1	0
D	0	0	0	1	1	1	1	1
A	0	0	0	0	1	0	1	1

De nombreux ensembles de 5 points de X pourraient être affectés à des sous-figures de 5 points de Y. Cependant, cette affectation doit être vérifiée aussi par l'algorithme décrit pour deux formes qui ont le même nombre d'éléments.

Par exemple, F C B D A de la forme X est similaire au sous figures de Y suivantes :

G B E C D, G A E C D, G B E H D, G A E H D, B E H C F, G B H C D, G A H C D, G B E F D.

Naturellement, on peut appliquer le procédé à une forme X elle-même pour trouver toutes les sous-figures équivalentes $Z \subset X$. Par exemple, dans la forme X de la figure IV.18.X, les formes partielles F C B D A et F C E D A sont similaires.

Soient X et Y deux formes, déterminées au hasard, mais ayant un nombre donné de points : $|X| = n_1$ et $|Y| = n_2$. Il est intéressant par ce procédé, de déterminer les sous-formes équivalentes. On trouve expérimentalement que le nombre Z de sous formes équivalentes reste inférieur à un seuil. Par exemple, si $n_1 = 6$ et $n_2 = 8$, le tableau ci-après donne les résultats du test ; la première ligne indique la taille des sous-figures considérées.

6	5	4	3	2	1
0	0	120	572	420	48
0	5	144	568	420	48
0	21	249	544	420	48
0	32	266	536	4 20	48
0	18	180	573	420	48

Les cinq autres lignes donnent le nombre Z des sous-formes équivalentes pour 5 couples différents de formes X et Y générées aléatoirement.

Si $n_1 = n_2 = 10$, avec 8 couples de formes essayées, on n'a pas trouvé de sous-figure équivalente pour $Z \geqslant 8$. Pour $Z = 7$ on trouve des sous-figures pour 3 couples sur 8. Pour $Z < 7$ de nombres figures partielles sont détectées.

Ces résultats sont à comparer à ceux obtenus par filtrage linéaire dans des images échantillonnées et présentant du bruit (cf. J. C. SIMON et J. CAMILLERAPP [ ]). Au delà d'un certain seuil on peut considérer que la forme observée est du "bruit".

Ici, encore on peut invisager l'introduction d'un procédé de parenthésage afin de permettre un certain degré de liberté supplémentaire dans la comparaison.

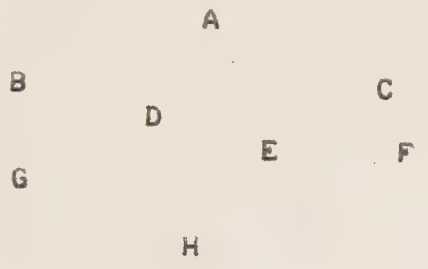


FIG. IV.13.X

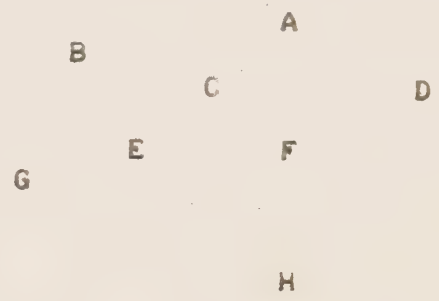


FIG. IV.13.Y

A B G D H E F C
C A B D G E H F
B A G H E D F C
D C A B G H E F
F C A B D E G H
E C A D B G H F
G F E C D A B H
H F C E A D B G

FIG. IV.14.X

A B G E1C2H1F2D
B A G E H F C D
D A B C E G F H
C A B G E H F D
F D A C B E G H
E D C1A2B G H F
G F D E C A B H
H D F1A2C B E G

FIG. IV.14.Y

```
GAHEB
 ENVELOPPE  EABGH
DCFE
 ENVELOPPE  FCDE
CDEAFBGH
 ENVELOPPE  CABGHF
HGFACB
 ENVELOPPE  CABGHF
EHADF
 ENVELOPPE  FADH
CEF
 ENVELOPPE  FCE
```

FIG. IV.15

SELECTED PØINTS ØF FIG. 1-X, FOLLOWED BY THE CONVEX ENVELOPE POINTS .

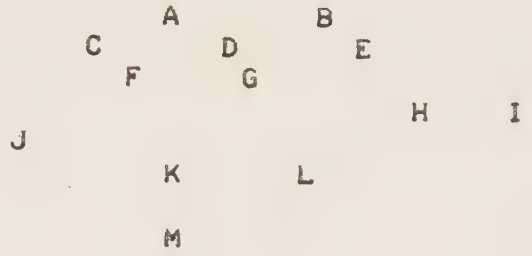


FIG. IV.16

RECONSTITUTION DE FIGURES

FIG. IV.17

FIG. IV.18.X

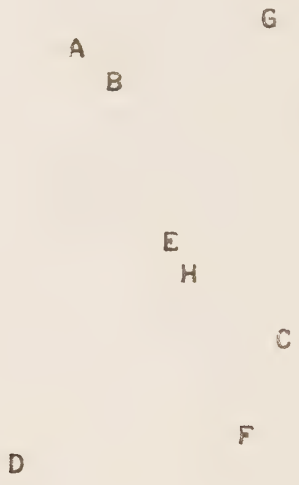


FIG.IV.18.Y

FIG. IV.20

FIG. IV.19

RECØNSTITUTION DE FIGURES

"LISTINGS" DES PROGRAMMES

```
"HDG    RECONNAISSANCE D'UN ENSEMBLE DE POINTS DU PLAN
"FOR,IS RECO
      PARAMETER NP=10,ND=45,NFI=10,NCA=45,NCB=70
      DIMENSION CHI(9),AB(NP),OR(NP),X(NP),Y(NP),D(NFI,ND),TAB(NCA,NCB),
     1P(NFI,NFI)
      COMMON NB,NL,DB,NITER,NCOMP,M(NP,NP),OF(ND),EF(ND),DF(NP),
     1OT(ND),ET(ND),DT(ND),IL(NP),IDI(ND),IFI(ND),KF
      INTEGERO(NFI,ND),E(NFI,ND),OF,EF,OT,ET
      READ 33,(CHI(K),K=1,9)
   33 FORMAT(80A1)
C    GENERATION ALEATOIRE DES FIGURES
      READ 1,NH1,NH2
      READ 1,NB,NFB,NFV,IDB,IDV,NITER
    1 FORMAT(25I3)
      DB=IDB
      DV=IDV
      I=0
      ITAB=IDB+3+2*IDV
      JTAB=10*ITAB/6
      NCOMP=0
      P2I=8.*ATAN(1.)
      PRINT 44,NH1,NH2,IDB,IDV
   44 FORMAT(' FIGURES GENEREES A PARTIR DES NOMBRES NH1=',I3,' NH2=',I3
     1/' COMPRRISES DANS LE CARRE DE COTE ',I3,/' RAYON DE VARIATION AUT
     2OUR DE CHAQUE POINT',I3)
C DETERMINATION D'UNE FIGURE DE BASE
      DO 20 MM=1,NFB
      DO 21 K=1,NB
      AB(K)=ALEAHU(NH1,NH2)*DB+1.+DV
   21 OR(K)=ALEAHU(NH1,NH2)*DB+1.+DV
C    DETERMINATION DES VARIANTES
      DO 20 N=1,NFV
      IF(N-1)22,23,22
   23 DO 24 K=1,NB
      X(K)=AB(K)
   24 Y(K)=OR(K)
      GO TO 26
   22 DO 25 K=1,NB
      RO=ALEAHU(NH1,NH2)*DV
      THETA=ALEAHU(NH1,NH2)*P2I
      X(K)=AB(K)+RO*COS(THETA)
   25 Y(K)=OR(K)+RO*SIN(THETA)
C    CALCUL ET STOCKAGE DES DISTANCES
   26 CALL DIST(X,Y,DF,OF,EF)
      I=I+1
      DO38 J=1,NL
      D(I,J)=DF(J)
      O(I,J)=OF(J)
   38 E(I,J)=EF(J)
C    DESSIN DE LA FIGURE
      DO 27 K=1,ITAB
      DO 27 L=1,JTAB
   27 TAB(K,L)=1H
      DO 28 K=1,NB
      IX=X(K)
      IY=10.*Y(K)/6.
      TAB(IX,IY)=1H*
      IF(K-10)29,30,30
```

```
   29 TAB(IX,IY+1)=CHI(K)
      GO TO 28
   30 KK=K/10
      TAB(IX,IY+1)=CHI(KK)
      KKK=K-10*KK
      IF(KKK)31,32,31
   32 TAB(IX,IY+2)=1H0
      GO TO 28
   31 TAB(IX,IY+2)=CHI(KKK)
   28 CONTINUE
      PRINT34,I
   34 FORMAT(1H1' FIGURE',I3//)
      PRINT 35,(X(K),K=1,NB)
      PRINT 35,(Y(K),K=1,NB)
   35 FORMAT(20F6.2)
      DO 36 K=1,ITAB
   36 PRINT 37,(TAB(K,L),L=1,JTAB)
   37 FORMAT(1X,120A1)
   20 CONTINUE
      NF=I
      PRINT 41
   41 FORMAT(1H1,' COMPARAISON DES COUPLES DE FIGURES'///)
      DO 3 I=1,NF
C    CHOIX DE LA FIGURE I
      DO 4 K=1,NL
      OF(K)=O(I,K)
      EF(K)=E(I,K)
    4 DF(K)=D(I,K)
      DO 3 J=1,NF
      IF(J-I)8,3,8
C    CHOIX DE LA FIGURE J
    8 DO 5 K=1,NL
      OT(K)=O(J,K)
      ET(K)=E(J,K)
    5 DT(K)=D(J,K)
C    DETERMINATION DE P2
      CALL DICHO(P2)
C    REAPPEL DE COMPAR POUR P2
      CALL COMPAR(P2,INDIC)
      P(I,J)=P2
      PRINT 13,I,J,P2,(IL(K),K=1,NB)
   13 FORMAT(' COUPLE',I3,1H,,I2,'  P2=',F8.4,3X,25I3)
      IF(INDIC)3,15,3
   15 PRINT 40
   40 FORMAT(1H+,80X,'AFFECTATION REALISEE')
    3 CONTINUE
      PRINT 11,(I,I=1,NF)
      DO 10 I=1,NF
   10 PRINT12 ,I,(P(I,J),J=1,NF)
   11 FORMAT(//' TABLEAU DES MARGES D ERREUR LIMITES'//3X,15I8)
   12 FORMAT(I5,15F8.4)
      PRINT7 ,NCOMP
    7 FORMAT(///'  NOMBRE DE COMPARAISONS',I5)
      STOP
      END
"FOR,IS DICHO
C   DICHOTOMIES SUCCESIVES
      SUBROUTINE DICHO(P2)
```

```
      PARAMETER NP=10,ND=45
      COMMON NB,NL,DB,NITER,NCOMP,M(NP,NP),OF(ND),EF(ND),DF(NP),
     1OT(ND),ET(ND),DT(ND),IL(NP)
      INTEGER OT,ET,OF,EF
      P=-0.10
      P1=0
      P2=0
   10 P=P+0.10
      CALL COMPAR(P,INDIC)
      IF(INDIC)11,12,14
   11 P1=P
      GO TO 10
   12 IF(P)19,19,14
   14 P2=P1+0.10
      DO 16 K=1,NITER
      P=(P1+P2)/2.
      CALL COMPAR(P,INDIC)
      IF(INDIC)17,18,18
   17 P1=P
      GO TO 16
   18 P2=P
   16 CONTINUE
   19 RETURN
      END
"FOR,IS COMPAR
C    COMPARAISON DES FIGURES POUR P DONNEE
      SUBROUTINE COMPAR(P,INDIC)
      PARAMETER NP=10,ND=45
      COMMON NB,NL,DB,NITER,NCOMP,M(NP,NP),OF(ND),EF(ND),DF(NP),
     1OT(ND),ET(ND),DT(ND),IL(NP),IDI(ND),IFI(ND),KF
      INTEGER OT,ET,OF,EF
      DIMENSION IC(NP),LIG(ND)
      NCOMP=NCOMP+1
      DO 7 I=1,NB
      DO 7 J=1,NB
    7 M(I,J)=1
C    PARENTHESAGE DE LA FIGURE T
      CALL PARENT(IDI,IFI,P,DT)
C    CONSTRUCTION DE LA MATRICE D'AFFECTATION
   82 DO 54 I=1,NL
      KF=OF(I)
      CALL AFFECT(I)
      KF=EF(I)
   54 CALL AFFECT(I)
C    REDUCTION LOGIQUE
      ITEST=0
      NTEST=0
   38 NEP=0
      DO 338 II=1,NB
      IL(II)=0
  338 IC(II)=0
  438 DO 32 I=1,NB
      IF(IL(I))39,39,32
   39 CALL NETOIL(I,NE,IDE)
      IF(NE-1)31,30,32
   30 IF(IC(IDE))73,71,73
   73 IF(IC(IDE)-I)31,32,31
   71 IL(I)=IDE
```

```
      IC(IDE)=I
      NEP=NEP+1
      DO 40 K=1,NB
   40 M(K,IDE)=0
      M(I,IDE)=1
   32 CONTINUE
      DO 35 J=1,NB
      IF(IC(J))35,33,35
   33 CALL NETOIC(J,NE,JDE)
      IF(NE-1)31,34,35
   34 IF(IL(JDE))41,42,41
   41 IF(IL(JDE)-J)31,35,31
   42 IC(J)=JDE
      IL(JDE)=J
      NEP=NEP+1
      DO 43 K=1,NB
   43 M(JDE,K)=0
      M(JDE,J)=1
   35 CONTINUE
C    TESTS D'ARRET
      IF(NEP-NB)401,400,401
  400 IF(NTEST)98,98,61
  401 IF(NEP-ITEST)98,98,122
  122 ITEST=NEP
      GO TO 438
   98 IF(NEP-NTEST)1000,1000,118
C    REDUCTION ORIGINE -EXTREMITE
  118 DO 20 I=1,NL
      IOF=OF(I)
      IEF=EF(I)
      IF(IL(IOF))21,21,23
   21 IF(IL(IEF))20,20,22
   22 ISA=IEF
      ISC=IOF
      GO TO 25
   23 ISA=IOF
      ISC=IEF
   25 IDP=IDI(I)
      IFP=IFI(I)
      IIL=IL(ISA)
      DO 51 K=1,NB
   51 LIG(K)=0
      DO 26 J=IDP,IFP
      IOT=OT(J)
      IET=ET(J)
      IF(IOT-IIL)28,27,28
   28 IF(IET-IIL)26,29,26
   27 IIT=IET
      GO TO 50
   29 IIT=IOT
   50 LIG(IIT)=1
   26 CONTINUE
      DO 420 J=1,NB
      IF(M(ISC,J))420,420,351
  351 IF(LIG(J))52,52,420
   52 M(ISC,J)=0
  420 CONTINUE
   20 CONTINUE
```

```
      NTEST=NEP
      GO TO 38
C   DECISION DANS LA COMPARAISON
 1000 INDIC=1
      RETURN
   61 INDIC=0
      RETURN
   31 INDIC=-1
      RETURN
      END
"FOR,IS DIST
C   CALCUL ET CLASSEMENT DES DISTANCES
      SUBROUTINE DIST(X,Y,D,O,E)
      PARAMETER NP=10,ND=45
      INTEGER O(ND),E(ND)
      DIMENSION D(ND),X(NP),Y(NP)
      COMMON NB,NL
      K=0
      NNB=NB-1
      DO 1 I=1,NNB
      II=I+1
      DO 1 J=II,NB
      K=K+1
      D(K)=SQRT((X(I)-X(J))**2+(Y(I)-Y(J))**2)
      E(K)=J
    1 O(K)=I
      NL=K
      NNL=NL-1
      DO 3 I=1,NNL
      II=I+1
      DO 3 J=II,NL
      IF(D(I)-D(J))2,3,3
    2 DD=D(I)
      D(I)=D(J)
      D(J)=DD
      JJ=E(I)
      E(I)=E(J)
      E(J)=JJ
      JJ=O(I)
      O(I)=O(J)
      O(J)=JJ
    3 CONTINUE
      RETURN
      END
"FOR,IS AFFECT
C   AFFECTATION AU SOMMET KF DE F DES SOMMETS ENTRE IDP ET IFP DE I
      SUBROUTINE AFFECT(I)
      PARAMETER NP=10,ND=45
      COMMON NB,NL,DB,NITER,NCOMP,M(NP,NP),OF(ND),EF(ND),DF(NP),
     1OT(ND),ET(ND),DT(ND),IL(NP),IDI(ND),IFI(ND),KF
      INTEGER OT,ET,OF,EF
      DIMENSION LIG(NP)
      IDP=IDI(I)
      IFP=IFI(I)
      DO 11 MM=1,NB
   11 LIG(MM)=0
      DO 12 K=IDP,IFP
      KOT=OT(K)
```

```
      KET=ET(K)
      LIG(KOT)=1
   12 LIG(KET)=1
      DO 17 K=1,NB
      IF(M(KF,K))13,17,13
   13 IF(LIG(K))17,14,17
   14 M(KF,K)=0
   17 CONTINUE
   16 RETURN
      END
"FOR,IS NETOIL
C   NOMBRE DE 1 DE LA LIGNE I
      SUBROUTINE NETOIL(I,N,J)
      PARAMETER NP=10,ND=45
          ON NB,NL,DB,NITER,NCOMP,M(NP,NP)
      N=0
      DO 1 K=1,NB
      IF(M(I,K))1,1,2
    2 N=N+1
      J=K
    1 CONTINUE
      RETURN
      END
"FOR,IS NETOIC
C   NOMBRE DE 1 DE LA COLONNE I
      SUBROUTINE NETOIC(I,N,J)
      PARAMETER NP=10,ND=45
      COMMON NB,NL,DB,NITER,NCOMP,M(NP,NP)
      N=0
      DO 1 K=1,NB
      IF(M(K,I))1,1,2
    2 N=N+1
      J=K
    1 CONTINUE
      RETURN
      END
"FOR,IS PARENT
C   PARENTHESAGE
      SUBROUTINE PARENT(IDI,IFI,P,DF)
      PARAMETER NP=10,ND=45
      COMMON NB,NL,DB
      DIMENSION IDI(ND),IFI(ND),DF(ND)
      DR=P*DB
      I1=1
      DO 10 I=1,NL
      IF(DF(I)-DF(I1)-DR)11,11,15
   11 DO 12 J=I1,NL
      IF(DF(I)-DF(J)-DR)13,13,14
   13 IDI(J)=I
   12 CONTINUE
      DO 16 K=I,NL
   16 IFI(K)=NL
      GO TO 82
   14 I1=J
      IFI(I)=I1-1
      GO TO 10
   15 IFI(I)=IFI(I-1)
   10 CONTINUE
```

```
   82 RETURN
      END
"XQT,N
123456789
  5  7
 10  3  3 30  3  4
"FIN
```

```
PAGE   1

// JOB

LOG DRIVE    CART SPEC    CART AVAIL   PHY DRIVE
  0000         3333         3333         0000

V2 M07   ACTUAL  8K  CONFIG  8K

// XEQ REPRO      1
```

Bibliographie relative au chapitre IV

[1] SHEPARD R. N. = " The analysis of proximities. Scaling with an unknown distance function." I psychometrica, June 1962, vol. 27, n°2.

[2] BENZECRI : "Cours sur l'Analyse des Correspondances"(ISUP).

[3] SOKAL R. R. and SNEATH P. H. A. : "Principles of numerical Taxonomy". San Francisco and LONDON (1963).

[4] M. EDEN : "On the formalization of Handwriting". Proceeding of Symposia in Applied math, 12, pp. 83-88, Am. Math. Soc., Providence R. I., 1961.

[5] M. EDEN : "Handwriting and Pattern Recognition". IRE Trans. on Inf. Theory, 8, pp. 160-166, 1963.

[6] K. S. FU and P. H. SWAIN : "On syntactic Pattern Recognition". 3rd Symposia on Computers and Inf. Science, COINS 1969. Bal Harbour, Fla, Dec. 18-20, 1969.

[7] R. A. KIRSCH : "Computer Interpretation of English Text and Picture Patterns". IEEE. Trans. on Elect. Computers, EC-13, pp pp. 363-376, 1964.

[8] M. MINSKY : "Steps towards Artificial Intelligence". Proc. IRE, 49, # 1, Jan. 1961.

[9] R. NARASIMHAN : "Labeling Schemata and Syntactic Descriptions of Pictures", Information and Control, 7, pp. 151-179, 1964.

[10] J. L. PFALZ and A. ROSENFELD : "Web Grammars". Proc. of the Int. Joint Conf. on Art. Int. (IJCAI) May 1969.

[11] J. C. SIMON and J. CAMILLERAPP : "Recherche d'une Forme dans un Fond". C. R. Ac. Sc., Série A, 267, pp. 946-949, 16 Décembre 1968.

[12] J. C. SIMON, A CHECROUN and C. ROCHE : "Comparaison des Formes Indépendantes des Déplacements, Homothéties et petites Déformations". C. R. Ac. Sc., Séries A, 270, pp. 1 607-1 609, 15 Juin 1970.

[13] J. C. SIMON, A. CHECROUN and C. ROCHE : "A Method of Comparing two Patterns Independant of possible Transformations and small Distorsions". IEEE Symposium on syntactic pattern recognition. Chicago, Oct. 1970. To be published in Pattern Recognition Journal.

[14] J. C. SIMON and A. CHECROUN : "A pattern Linguistic Analysis Invariant in Plan Transformations". IJCAI, Symposium on Artificial Intelligence. Londres (Sept. 1971).

[15] K. S. FU : "Sequential Methods in Pattern Recognition and Machine Learning". New York. Academic Press (1968).

Académie de Paris

UNIVERSITE de PARIS VI
11, Quai Saint Bernard - Paris 5

SERVICE DE LA SCOLARITE
Bureau des Enseignements du 3e Cycle

tél. 336 25.25 et 325.12.21

PERMIS D'IMPRIMER

DOCTORAT : DOCTORAT ES SCIENCES

THESE DE : SCIENCES MATHEMATIQUES

NOM : Monsieur CHECROUN

Prénoms : Alain

né le 20/9 1936 à ALGER Dépt : ALGERIE

1ère THESE : "Traitement et reconnaissance des formes par un ordinateur. Application à la cartographie électromagnétique."

2e THESE (Propositions faites par l'Université) : "Problèmes de Cauchy abstraits et semi-groupes distributions.

Vu et Approuvé

PARIS, le 20 mars 1972

Le Président

de l'Université de PARIS VI

A. HERPIN